Beyond Deployments and Testbeds: Experiences with Public Usage on Vehicular WiFi Hotspots

Joshua Hare, Lance Hartung, and Suman Banerjee
Dept. of Computer Sciences, University of Wisconsin-Madison
Madison, WI, USA
{hare, hartung, suman}@cs.wisc.edu

ABSTRACT

We describe our experiences with deploying a vehicular Internet access service on public transit buses. Our system, called WiRover, has been running on these buses since April 2010 for about 18 months till date providing a WiFi hotspot to which bus passengers can connect. As of Dec. 1, 2011 we have observed 17,567 unique client devices connect to the WiRover system. These devices have downloaded more than 337.53 GB and uploaded 48.19 GB of content through our system. Since the initial deployment, the buses have logged more than 9,331 hours of operation and have traveled over much of the northern Midwest of the United States. Through this paper we provide different insights acquired in deploying and running a continuous service with real users, present various user and usage characteristics in these systems, discuss various design and management strategies for these networks, and explore different network traffic optimizations possible.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless Communications; C.2.3 [**Network Operations**]: Public Networks, Network Management

General Terms

Experimentation, Measurement, Performance

Keywords

Network Diversity, Vehicular Connectivity, Wireless, WiRover

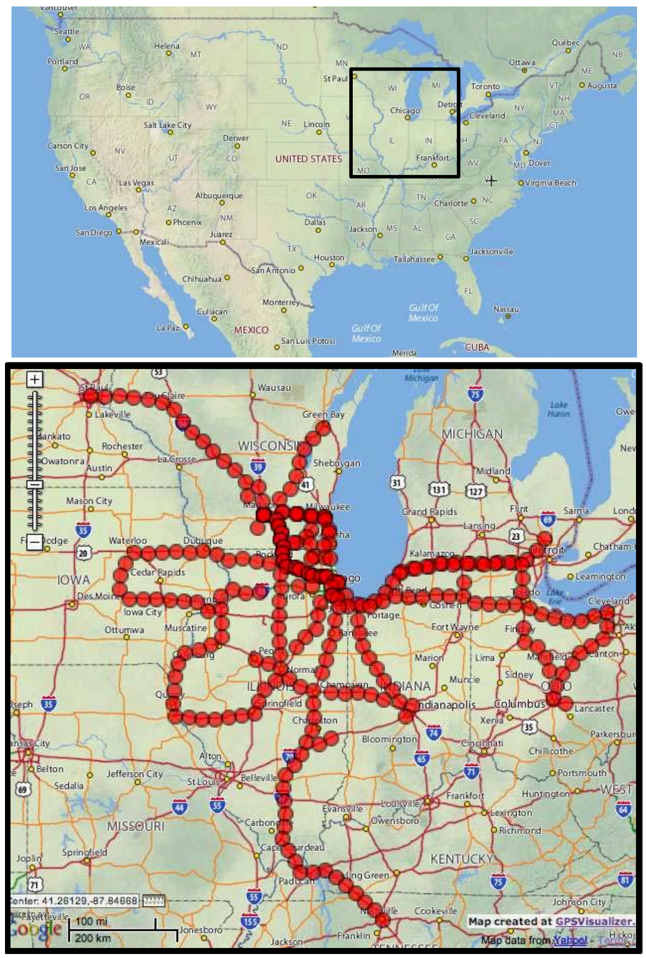

Figure 1: The 5 public buses on which we have installed our WiRover system have traveled over much of the northern Midwest region of the United States (the map covers 988763 sq km).

1. INTRODUCTION

Internet connectivity for moving vehicles and their passengers is an important and challenging problem domain. It is now becoming common for public transportation buses to offer WiFi services to passengers [25]. The growth of WiFi connectivity in mass transportation is causing vehicular networking to become ubiquitous (famously Amtrak trains [1], Greyhound buses, and Google buses [24]). Despite the rapid growth in vehicular Internet connectivity, little is known about the network performance, usage characteristics, and operational challenges of providing such connectivity in vehicular environments. For instance, given the widespread availability of data plans on our mobile phones, would users even care to access the Internet through such services? If users do indeed use these services, how much can common

MobiSys'12, June 25–29, 2012, Low Wood Bay, Lake District, UK.
Copyright 2012 ACM 978-1-4503-1301-8/12/06 ...$10.00.

caching and compression techniques help if deployed on a single public bus carrying a few tens of users? How should one deal with outages of individual cellular networks when providing Internet connectivity to such public vehicles?

Over the last few years, we have designed and refined a vehicular Internet connectivity system, we call *WiRover* and have installed this system on a diverse set of vehicles based out of Wisconsin, USA. More specifically, our WiRover system has been installed on more than 10 different public buses, ambulances, and personal vehicles. In this work, we focus on 5 particular vehicles which happen to be passenger buses, and share our operational experience and learnings from deploying and operating an Internet service on these buses for more than 18 months. These 5 buses consist of 2 city transit buses and 3 coach buses, which have covered much of the northern Midwest of the United States, as shown in Figure 1. The WiRover system provides bus passengers with a free Internet connection by creating a WiFi hotspot within the bus. Passengers connect to this free service to obtain Internet services. In exchange for this free service, our terms of service allow us to monitor network traffic characteristics (anonymized, and in aggregate) for research purposes. The goal of our study has been to use our deployment to understand various issues and challenges in providing a continuous and robust service to all users in public transit systems.

When we first deployed our system we were unsure as to how many smart phone users might connect to our system. After all, each smart phone user already has a mobile cellular data plan and may not need to use the services of WiRover. Hence, to be successful as a popular vehicular WiFi hotspot, we needed to ensure that the experience through WiRover is superior to that of the individual's own smart phone. Further, our system needed to be robust and highly available so that users can begin to depend on it in their daily commutes or occasional long distance bus rides.

We believe that we have been quite successful in being able to attract and retain users. This is particularly apparent when we observe the large number of various smart phone users who continuously access Internet services through our WiRover system, despite having data plans readily available on their mobile phones.

We believe this is due to three reasons. First, WiRover provides a faster Internet download experience for users through a number of optimizations built into the WiRover system, particularly the ability to aggregate the bandwidth from multiple networks. Figure 2, shows that WiRover is able to provide an average speedup of 1.37 over a single interface. Second, it provides reliable connectivity to users by utilizing its multi-network design. The two cellular networks that we use commonly in each bus (let us call them NetA and NetB) , individually have an availability of about 77.96% and 80.88% respectively across the entire area under study. The lower numbers reflect poorer coverage on certain rural highways and cities, and also reflect periods of time when these individual networks might be too slow to provide a useful Internet service. However, in our typical dual-network solution mounted on the buses, the two networks together increase overall availability to 94.37%. Finally, we also feel that providing a free service is enough to attract curious passengers.

Ensuring a reliable Internet connection has not been without its challenges. Hardware failures are challenging to debug since access to the hardware platform first requires access to the vehicle. Software bugs may remain dormant for days or months, making them difficult to identify and reproduce. Throughout this paper, we share the challenges we have faced and the mechanisms we have adopted to overcome the challenges of running similar deployments.

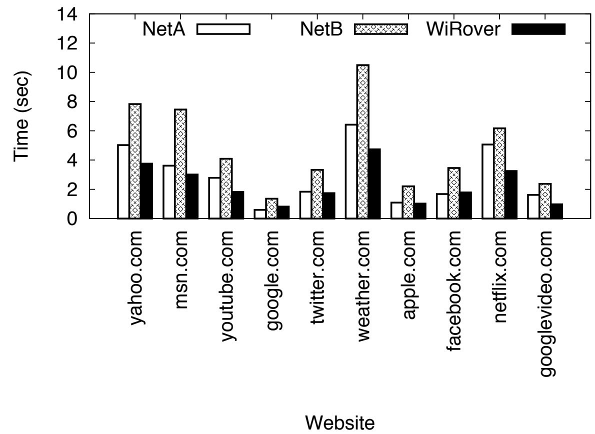

Figure 2: The website download times to ten of the most popular visited websites. WiRover provides an average speedup of 1.37 over a single NetA interface.

1.1 Contributions

The vehicular environment introduces many challenges that are not present in stationary networks. Users connect to our WiFi hotspot just as they would at a coffee shop and some users may have the expectation that their experience will be similar in both environments. However, in the vehicular environment it is necessary to use wireless back-haul networks and the fluctuating performance and link failures of these networks must be contended with.

In the past there have been many notable projects that presented innovative systems to improve Internet connectivity into vehicles to study different topics of interest, e.g., delay-tolerant networking (DieselNet) [23, 26, 3], to survey status of different WiFi hotspots across a metropolitan area (CarTel) [7, 9], to collect road condition information through vehicle-mounted sensors (Pothole Patrol) [10], to collect various types of wireless measurements across a wide-area (WiScape) [22], and to design techniques to improve achievable throughputs to moving vehicles (ViFi, Wiffler, Pluribus) [5, 4, 14] as well as others [8, 12, 15, 20]. This paper does not focus on any of these individual problem domains, including fine-tuning algorithms that improve end-to-end throughputs to moving vehicles. Instead the goal of this paper is to share our experiences in running a vehicular Internet access system on multiple city and long distance buses for a period of more than 18 months *as a service with thousands of real users depending and continuously using it* and the ensuing lessons learned from this effort. While all the above cited efforts worked significantly with vehicular deployments, some as very long running efforts, we believe that none of these prior efforts provided a continuous Internet access system to the multitude of bus passengers on the move that we are providing daily and are reporting in this

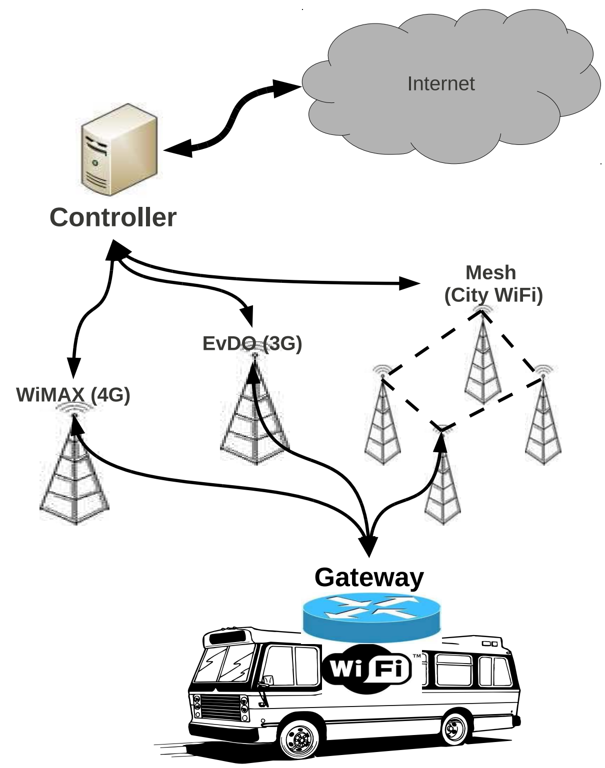

Figure 3: The WiRover system can connect to any wireless back-haul channel to provide Internet connectivity locally to bus passengers.

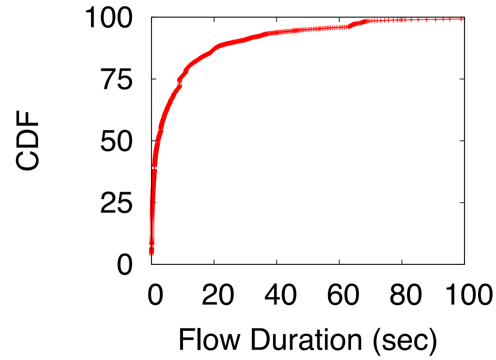

Figure 4: The CDF of flow durations as observed over the past 18 months show that more than 85% are less than 20 seconds.

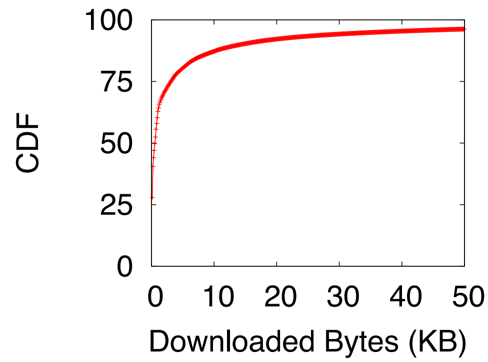

Figure 5: The CDF of flow size as observed over the past 18 months show that more than 62% of flows download less then 1 KB.

paper. While building vehicular testbeds is always hard, we have found that running a continuous service operating on these vehicles on which users can depend upon is a bigger challenge. While small failures that impact individual nodes in a testbed can be addressed in an opportunistic fashion, and often allow experimenters to ignore such nodes for small periods of time, failures in our system lead to unhappy users and often required us to quickly detect and repair such occurrences.

Summarizing, this paper makes the following contributions:

- Experiences in designing and implementing a simple yet sustainable vehicular Internet service, deploying and operating it for more than 18 months across both local and long distance public transit buses, and in managing this system to preserve uptime (Section 2 and 3).

- Characteristics of user behavior and usage, when accessing Internet services from these vehicles, including some differences in behaviors across the passengers of city transit and long distance buses (Section 4).

- Some potential optimizations to improving user experience and their potential usefulness in our system, especially in the applicability of techniques such as Packet Caches for reducing traffic across "expensive" cellular links. (Section 5).

- Exploration of the various types of failures experienced by our system, including those due to spotty and variable network coverage, their impact on user traffic, and our approaches to mitigate or manage them (Section 6).

- Discussion of some open research challenges while running our deployment over the past 18 months (Section 8).

2. DESIGN

Our WiRover system builds on certain basic design tenets of the MAR system [21]. The MAR system proposed a vehicular gateway which connected to multiple cellular networks to increase system bandwidth through bandwidth aggregation. These multiple cellular networks further increased the system's diversity and thus increased the reliability. We comment on some of our design choices and differences from the MAR system in this section, including some of our software scaffolding to implement a long running vehicular Internet service. They include: (i) our encapsulation structure and a simple flow scheduling approach, (ii) flow migration techniques in order to mitigate network failures and outages, (iii) a continuous active and passive network monitoring subsystem, and (iv) related traffic optimizations.

Encapsulation and simple flow scheduling: As shown in Figure 3, our WiRover system consists of a gateway node and a controller node (the proxy server). The gateway creates a single encapsulation tunnel across all the available wireless back-haul networks between the gateway and controller nodes. Traffic is modified through network address translation (NAT) at both the gateway and the controller nodes. While this encapsulation structure allows for partitioning of packets of an individual flow across multiple cellular paths, we chose a simpler alternative (as in MAR) of mapping individual flows to a single cellular path only, based on dynamic weights assigned to each path. (The only exception to this rule was during failures and slowdowns in individual cellular paths as described in the next section.) This design choice was pragmatic since in typical busy periods such flow mapping was adequate to extract all advantages of bandwidth aggregation in our system. Further, our flow logs collected over the last 18 months show that most flows are small and short-lived (Figure 4 and Figure 5), and the complexities of flow partitioning can easily outweigh the advantages in such cases.

Flow migration during link failures: Each cellular network has its own spatio-temporal fluctuations occasionally leading to significant slowdowns and outages. Since our flows are mapped entirely to a single cellular path, we chose to implement a simple flow migration scheme when such slowdowns or outages are detected. These events are detected on a cellular path triggering future packets for all flows in the failed path to be mapped to a new path. The receiver node (controller or gateway depending on the direction of the flow) detects the onset of a flow re-map and performs transient re-order management for that flow, i.e., out-of-order packets from a re-mapped flow are held for a dynamic period of time (the period is governed by the latency and bandwidth characteristics) to minimize end-to-end reordering of these packets.

Network performance measurement subsystem: The measurement subsystem monitors the performance of the different cellular back-haul paths. If the performance on a particular network drops below configurable thresholds (such as, average throughput below 20 Kbps, and at least $4\times$ lower than at least one other alternate path, or recent loss rates in excess of 50%), then the measurement subsystem can mark that cellular path as having failed. Of course, if a network interface card itself reports being disconnected from the network, that cellular path is also marked as failed. Flows currently mapped to these paths are then migrated to other alternate paths. To keep these measurements current, our measurement subsystem estimates achieved throughputs, latencies, and loss rates using an exponentially weighted moving average (EWMA) scheme. Samples for these metrics are obtained in two ways: (i) by passively observing ongoing user traffic mapped to the different cellular paths and (ii) in absence of recent observations, by generating periodic active measurements. The frequency of active measurements are also based on various statistics of historical measurements collected at different locations and at different times, insights for which were developed in our prior WiScape work [22].[1] We have observed that in busy periods of user activity, active measurements are rarely required.

Traffic optimizations: We have also experimented with the deployment of various caching and compression schemes in the gateway and the controller respectively. Such optimization can be extremely important to a system that utilizes expensive, low-bandwidth wireless network technologies, such as our cellular paths.

Remote management of a vehicular Internet service: Detecting problems and failures efficiently and swiftly is, perhaps, the most important challenge of operating a continuously running vehicular Internet service. Without specific safeguards, many problems that occur on the buses plying on local and long distant routes can go undetected for days and weeks. For instance, the mere fact that we can "ping" a gateway might hide the fact that the WiFi component has failed to boot up. Further, when problems are detected we need a systematic method to track down the relevant vehicle and be able to fix them in an expedited manner. The following are some of the current strategies in place based on our experiences.

- *GPS-based continuous tracking at controller:* Each gateway periodically reports its current location to our controller. Any time a gateway disappears from our controller's logs for an extended period of time, our system first checks its last known location. If the last known location is one of the designated garages for the vehicle, then we assume that the bus probably parked in the garage and went offline in a graceful manner. Often, individual buses can be garaged for a few days as they undergo servicing and repairs. If the last known location was not one of the garages, then it is a signal for likely problems encountered. In such cases, we have to contact the bus operator expeditiously to enquire about the whereabouts of the vehicle.

- *Forced reboots and automated log offloads:* In certain cases, a gateway might be unable to connect to the controller. This can happen due to poor connectivity across all cellular paths, e.g., it happens due to shielding effects in certain locations in the bus depots. Other times, it could be manifestation of software bugs that have not yet been fixed, or have been introduced in a new revision. If the gateway is unable to connect to the controller for a configurable period, we force a system re-boot. However, to understand the nature of the problem, we attempt to log all critical level events in the local hard disk in the gateway, and periodically offload these logs through the cellular interfaces when connectivity is available. This allows us to build detailed records of performance failures for future analysis.

- *Tracking usage activity:* If a gateway correctly reports to the controller, then we monitor the total user traffic that is transiting through our system and the number of users currently observed to be connected. If the number of users is zero for extended periods of time, despite the vehicle being observed to be moving, we match it against known routes. If it is on at least some designated route, then such a scenario can indicate a problem with the WiFi subsystem. We have SSH access into each gateway, and in many cases we can reset the WiFi interface if it has gone into some bad state. In case such repairs do not seem to work (in one instance, we observed that the gateway's hard disk was completely full which caused some associated malfunction), we use the GPS to determine when the buses are likely to pass by stops near our university campus. In our early months, we used to

[1]The WiScape project used a hardware platform similar to WiRover to collect measurements, however, the WiScape project did not provide WiFi service to passengers.

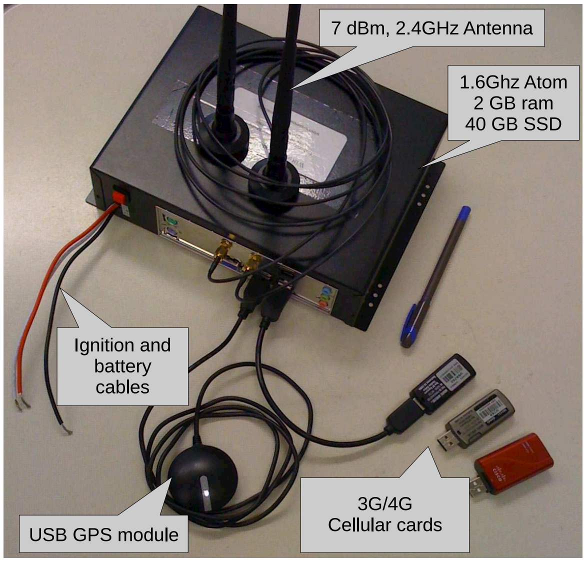

Figure 6: Our current version of our hardware setup consists of an off-the-shelf hardware platform outfitted with a power supply designed for vehicular settings and multiple wireless network modems as well as a GPS receiver.

board the bus during the stop durations and try some quick tests and fixes to revive the malfunctioning units.

- *Keeping backup units handy:* In some cases we have less than 5 minutes to fix the gateway unit before the bus continues on its routes, and we quickly realized that trying to fix nodes during these brief stops is quite hard. For this reason, in the recent months, we always keep a number of fully-configured backup gateway units handy. These days, we simply replace the malfunctioning unit with a backup unit, and bring the malfunctioning one into our lab for more careful evaluation and testing. This approach has significant streamlined our repair process.

- *User reports:* Finally, we routinely get a significant volume of comments, suggestions, and even bug reports from our user population. Many of these have helped in solving ongoing problems in the early months of operation.

We believe that a combination of all these approaches has significantly enhanced the robustness and the performance of our system over our continuing period of operation.

3. IMPLEMENTATION

Over the past year, our particular implementation of the WiRover system has evolved to address multiple issues as they have arisen. The WiRover code base runs on standard off-the-shelf hardware platforms with Linux as a combination of user-level processes and some specific kernel modifications. Our initial hardware setup consisted of a power inverter, a low cost netbook, and a Linksys wireless router.

3.1 Vehicular Power and Gateway Setup

This hardware setup was designed such that the system would initialize when the vehicle was turned on and would *gracefully* shutdown when the vehicle was turned off. In the early days of our deployment, it was most convenient to use a power inverter, a low-cost netbook (serving as the gateway), and a separate wireless router unit. A netbook platform was chosen so that it could gracefully stop the WiRover processes including archiving all unsaved data by using its own battery, once power supply to it was cut off. This design prevented daily crashes of the software at each power down event.

April 27, 2010	Deployed 1st Metro bus (M942).
Aug 12, 2010	Deployed 2nd Metro bus (M007).
Sept 17, 2010	Deployed 1st Coach bus (C001).
June 15, 2011	Deployed 2nd and 3rd Coach buses (C002-003)).

Table 1: Over the past year and a half we have deployed the WiRover system on 5 buses.

When the vehicle ignition was turned on, power was supplied to the inverter which in turn powered the netbook and the wireless router. The netbook was configured to boot-up using the Wake-On-LAN (WOL) mode, and once the wireless router started up, it was configured to wake up the netbook by sending WOL packets. This is a setup that had worked well for our prior *testbed-only* operations [17].

In our new installations for our vehicular Internet service, we initially encountered numerous failures with this simple setup. After some investigation we realized that unlike our previous deployment in [17] where their system was installed on a hybrid powered bus, our system was running on traditional diesel powered buses. The key difference between these two buses is the fact that diesel powered buses rely heavily on an electric powered motor to initially start the diesel engine. During this vehicle startup period of time the voltage level of the bus drops as this electric motor is consuming a large amount of power to start the engine. This drop in voltage was significant enough that the bus' voltage levels would drop below the voltage required for the power inverter to operate.

This would in turn cause the power to the wireless router to cut out in the middle of its startup process and *reset the wireless router's configuration settings.* Eventually all of the configuration settings that we had entered were reset and the wireless router no longer transmitted the WOL packets to wake up the laptop.[2] This served as a caution to check against treating the power supply issue casually, since if a power failure occurs once the bus is on the road, it becomes terribly inconvenient to determine the problems and fix them.

Current Hardware Solution: In light of our discovery, we moved to a more resilient hardware setup that could operate well on all types of buses. We chose to use a power supply that is designed for vehicular environments and has enough capacitance to overcome the voltage drop experienced during an engine turn over.

The current gateway nodes (Figure 6) consist of an embedded computer running an Atom 1.6 GHz processor with 2 GB of RAM and a 40 GB solid state hard drive. We use solid

[2]As an aside, we had multiple discussions with some researchers of the ViFi project, who complained of similar issues with WOL packets not suitably working with their deployment in a reliable way, and we were unaware as to why they worked for us in [17] and not for them.

state hard drives due to their resilience to vibrations, which thus reduces the likelihood of drive failures. The hardware platform further supports an *integrated WiFi card* which is capable of operating in AP mode which we leverage to provide WiFi connectivity to bus passengers. This also made our system quite compact.

For wireless backhaul and connectivity, our gateway currently uses two USB-based cellular modems (one NetA and one NetB), as well as a WiMAX modem. The WiMAX modem connects to our own experimental WiMAX base stations located on the roof of the CS building on campus. The buses can use this WiMAX network only when it is passing through specific parts of our campus in range of this base station.

The controller is a rack-mounted 2.66 GHz Quad core CPU with 4 GB of RAM located in our lab, and hosts the WiRover controller software along with various MySQL databases to store various performance logs and measurement data.

3.2 Vehicular Deployment

Our public vehicular Internet service is currently installed on 5 buses (at different times over the last 18 months as shown in Table 1), all of which share a single controller node. The buses are operated by two separate organizations: 2 buses are operated by Madison Metro Transit and will simply be referred to as Metro where the other 3 buses are operated by Van Galder Bus Company and will be referred to as Coach buses. There are three different types of buses: standard diesel Metro bus (M942), hybrid Metro bus (M007), and 3 standard Coach buses (C001-003). Our ability to install hardware on more buses is currently limited by the monthly cost of cellular data services. (The bus operators have expressed willingness to pay for this service and discussions on this are still underway.)

For the Metro buses, each bus is randomly assigned a bus route at the start of the day. Throughout the day the bus is likely to serve multiple different routes. For the Coach buses, each bus predominately serves a Madison, WI to Chicago, IL route but often times these buses are on charter routes and have covered much of the northern Midwest of the United States.

3.3 Basic System Benchmarks

Prior to presenting some of the detailed performance and usage characteristics of our system, there are two system overheads that are worth pointing out both of which relate to the use of a centralized controller.

Routing overheads: Our encapsulation tunnel forces all traffic to pass through our centralized controller. While it is possible to allow user traffic to flow directly to the Internet without touching the controller, it provides us two advantages: (i) flow migration in instances of individual cellular network failures, and (ii) ability to observe, manage, and configure all traffic from a convenient central location.

While this structure implies additional latencies on the path for users, given the high cellular latencies, the additional overhead is not overwhelming. Figure 7 shows the results of ICMP ping measurements (we found ICMP latencies to be representative of UDP and TCP latencies) to some popular web destinations that do not filter such packets, when using the cellular network with the least path latency

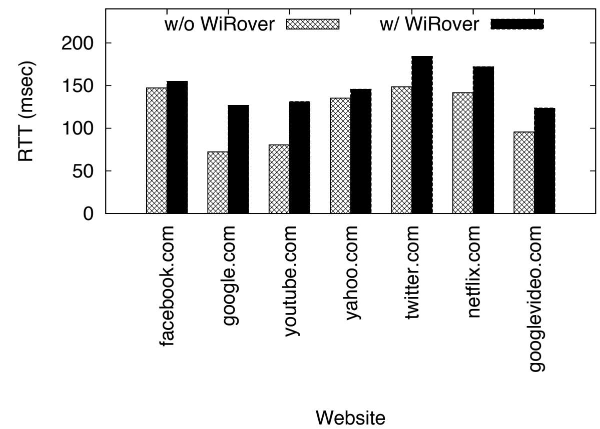

Figure 7: ICMP RTT measurements to some of the most popular web destinations when using the WiRover system as compared to each cellular network.

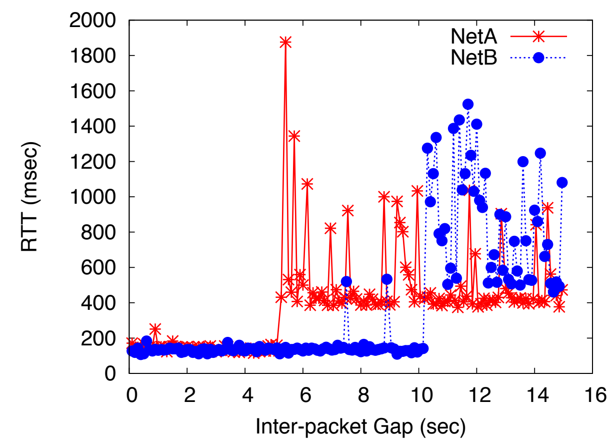

Figure 8: As the inter-packet gap is increased the cellular network re-installs network state increasing the packet delay.

in each case. We can see that the tunnel adds between 5 and 50 ms of additional latency for these ping packets. For most destinations the relative increase in latency is modest. The Google-controlled domains seem worst affected in a relative sense, possibly due to an efficient placement of the servers near the cellular networks.

Software overheads: Our benchmark of the WiRover system shows that the total software processing delay that is added to network packets is on the average of 320 μsec.

Tunnel overhead and fragmentation: The network tunnel adds 28 bytes of (IP and UDP) headers to data packets, which can increase the overall packet length beyond the maximum transmission unit (MTU) of the backhaul cellular networks. This reduces bandwidth efficiency on the end-to-end path. The bigger concern of this addition is the potential of packet fragmentation to packets that are already near the MTU.

Traditionally, such packet fragmentation can be avoided by running MTU discovery from the client device. However, we have found that most devices do not run MTU discovery and despite our efforts to provide the proper MTU values to clients via the dynamic host configuration protocol (DHCP) request some client devices choose to disregard this information. In such cases and as a last resort the WiRover system will overwrite the maximum segment size (MSS) field in TCP SYN packets which will reduce the maximum packet size for TCP packets. But packet fragmentation is still theoretically possible in our system for non-TCP packets, although we have not seen any occurrences.

NAT punching and other measurement issues: Various cellular networks also independently NAT traffic, and to ensure that our downlink path stays active and current, NAT punching was required. In absence of this, our system might inadvertently experience false failures. This is achieved through our periodic measurement subsystem. One additional issue addressed by our measurement subsystem shows up primarily during collecting latency estimates. It is well known that traffic inactivity in a certain cellular path can cause cellular base stations to remove bandwidth resources for such devices (in our case gateways). Arrival of new traffic requires synchronization between the cellular modem and the cellular base station [19], which causes additional delays for the first few new packets. Latency measurements on each path needs to be cognizant of these additional latencies. Figure 8, shows the round-trip time (RTT) delay for the second of two UDP ping packets as the inter-packet time gap is increased. The plot shows that packet RTTs in both networks go up significantly after some inactivity — the inactivity period is about 5 seconds for NetA and about 10 seconds for NetB.

4. USAGE CHARACTERISTICS

The WiRover system allows us to monitor usage and user characteristics of vehicular Internet services in the wild. These characteristics provide a valuable insight into how user's utilize such services, some differences in user behavior depending on travel routes, and as well as traffic characteristics that can be exploited for system optimizations. Further, the success of the system can also be monitored by presence and activity of repeat users who use our system. In this section, we first comment on usage volumes, including repeat users, the types of devices we see connect to our system, the popularity and types of traffic destinations, and draw specific conclusions from them.

4.1 Repeat Usage

As of Dec. 1, 2011 our WiRover system has provided Internet connectivity to 17,567 unique WiFi devices. However, it is important to understand how many of these devices have connected to our system multiple times. To accomplish this we define a *session* as a block of time where the device sends at least one byte of network traffic every 2 hours on a particular bus. If a time gap of greater than 2 hours is detected the data that the device transmits is grouped into two different sessions. Our notion of a session tries to capture all usage in a single bus ride as a single unit of activity. Since it is possible for a particular passenger to ride the same bus multiple times in one day, we have found that a 2 hour time period is sufficient to disambiguate the majority of these instances while still detecting passengers with sporadic usage on longer bus trips.

Figure 10, shows the growth of new devices and sessions over the duration of our deployment. The number of new devices grew more rapidly after July 2011 since only a short time before that we installed gateway nodes on 2 additional Coach buses. The number of repeat devices displays the total number of devices that have connected to the system at least twice since our data collection began. This demonstrates that despite the total number of repeat devices today is small (about 30%) of the total number of devices, repeat usage comprise a more significant fraction of our sessions (more than 55%).

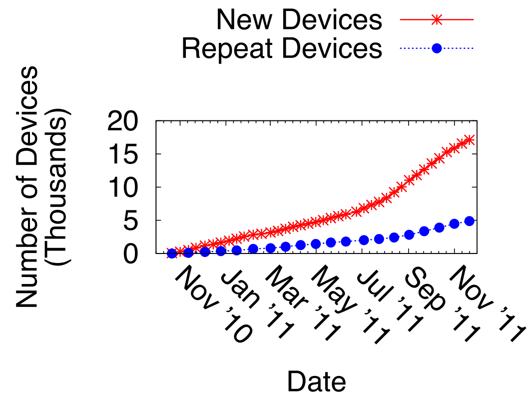

Figure 9: The growth of devices observed over time.

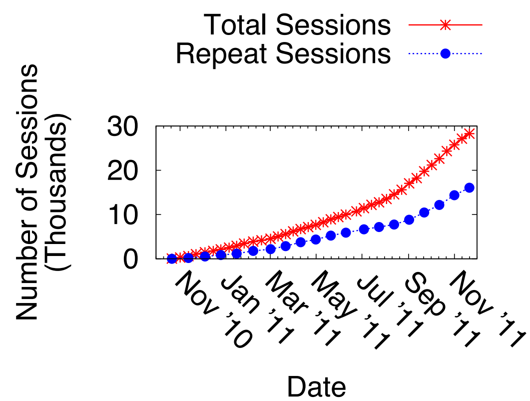

Figure 10: The growth of sessions observed over time.

We further find that 2427 of the 5079 repeat users are only observed on the Metro buses while only 761 repeat users are only observed on the Coach buses (1891 repeat users are observed on both). Despite the fact that we generally see a higher number of total users on the Coach buses (typically we see twice as many total users on the Coach buses), we observe that the Metro buses have almost twice as many repeat users as the Coach bus. This is likely because many Metro passengers ride the bus on a daily basis as they commute to and from work, where it is less likely many Coach passengers ride the bus with similar frequency.

4.2 Usage Characteristics

The usage characteristics for a particular device's session may be dependent on a number of factors such as the user's trip duration, the time of day, or even the type of device the

Device Platform	Metro buses Percent. of Sessions (%)	Avg Duration (Minutes)	Avg TX/RX Bytes	Coach buses Percent. of Sessions (%)	Avg Duration (Minutes)	Avg TX/RX Bytes
iPod	47.28	20.0	248 KB / 2.75 MB	44.44	83.0	825 KB / 9.56 MB
iPhone	36.69	16.6	189 KB / 1.62 MB	34.48	87.0	970 KB / 9.86 MB
Linux ARM	8.98	16.0	356 KB / 2.66 MB	8.45	93.2	1.29 MB / 15.2 MB
Apple Laptop	1.78	15.6	984 KB / 6.12 MB	1.67	48.8	5.42 MB / 55.3 MB
Windows Laptop	1.78	17.9	3.42 MB / 13.6 MB	1.67	48.1	3.67 MB / 27.5 MB
iPad	1.51	20.4	365 KB / 2.18 MB	1.42	76.1	1.94 MB / 17.0 MB
BlackBerry	0.80	5.83	26.4 KB / 338 KB	0.76	55.3	291 KB / 1.41 MB
Win Mobile	0.37	12.1	501 KB / 5.50 MB	0.35	54.6	199 KB / 1.07 MB
PSP	0.30	13.6	293 KB / 2.44 MB	0.28	64.3	297 KB / 6.34 MB
Nokia	0.29	8.0	89.2 KB / 384 KB	0.25	21.8	232 KB / 1.07 MB
Linux Laptop	0.23	9.24	749 KB / 4.54 MB	0.22	40.1	1.77 MB / 9.81 MB
Nintendo 3DS				0.02	97.6	1.04 MB / 10.7 MB

Table 2: Comparison of average usage characteristics for various device types. The overall average session duration for Metro buses is 18.13 minutes and was 53.23 minutes for Coach buses. The longest user session was observed on one of the Coach buses and was 645 minutes long (10.75 hours).

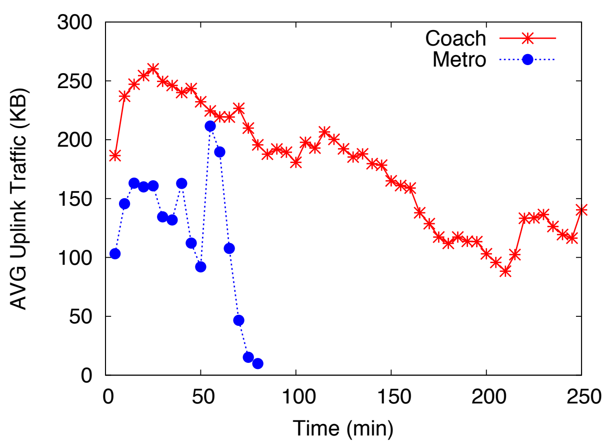

Figure 11: Average volume of uplink traffic over a user's session time.

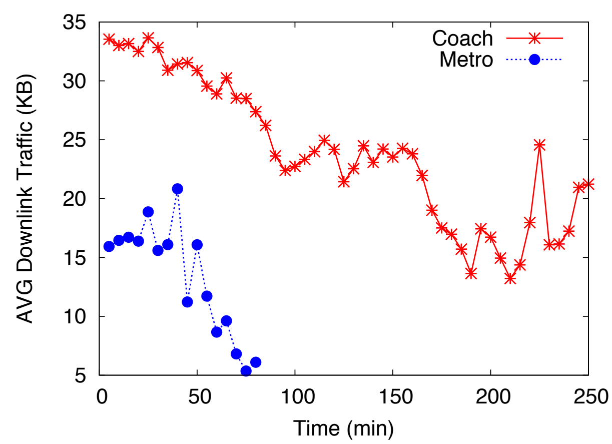

Figure 12: Average volume of downlink traffic over a user's session time.

user is interacting with. Table 2 shows the average session duration as well as the average number of bytes sent and received by various device types over the WiRover system. As can be seen in the table, there are differences in not only the device type but also based on the type of bus the passenger is on. Passengers on the Metro buses typically have shorter trip durations and this can be seen from the shorter average session durations and lower usage characteristics.

When we look at the average usage characteristics per session over time we notice that on average most usage is observed at the beginning of the session as shown in Figure 11 and Figure 12. For the Metro buses there is a sharp decrease in the average downlink traffic toward the end of user's session whereas usage on the Coach buses has a more graceful decrease. This may be due to the fact that users on the Metro buses use their devices right up to the point that they leave the bus. However, due to the longer durations of Coach bus trips, users may run out of battery life on their devices and scale back their usage over the trip duration.

Smart Phone Usage: Despite many smart phone users having dedicated cellular data plans we still find a large number of smart phone devices connecting to the system. As seen in Figure 13, smart phone devices consistently outnumber every other device category. We believe it is due to both the superior performance of the WiRover system (Figure 2), as well as the free WiFi-based service that help users conserve their data usage. Over the past few months we have seen over 3000 distinct devices across all of the buses with October 2011 yielding the highest single month total of 3787 distinct devices.

Other devices: We also observed a small number of unexpected devices that connected to our network. Examples include a Sony PlayStation Portable and a Nintendo 3DS. There were also a small number of feature phones that were observed to connect to our network, the maximum number of which never exceed 15 in a month.

4.3 Traffic Destinations

The user generated traffic that traverses our system is predominately HTTP traffic as is shown by Table 3. This is

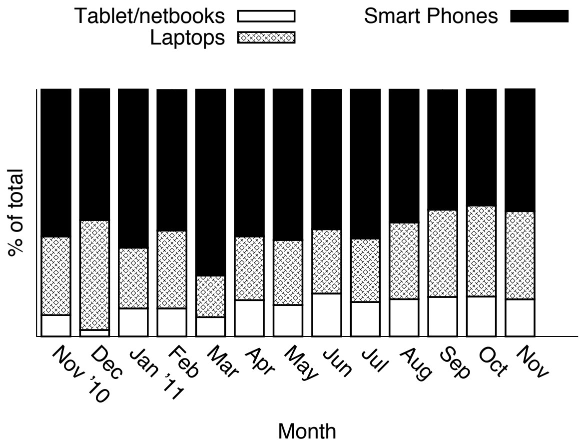

Figure 13: Distribution of various device type observed each month. Feature phones have been omitted from these results as they constitute less than 1% of devices for any given month.

Proto/Port	Bytes Received	Bytes Sent	Description
TCP 80	218 GB	19 GB	HTTP
TCP 443	17 GB	5.1 GB	HTTPs
TCP 1935	7.3 GB	227 MB	Flash RTMP
TCP 993	2.1 GB	307 MB	IMAPS
UDP 4500	870 MB	182 MB	IPsec NAT Trav.
TCP 22	832 MB	26 MB	SSH
TCP 995	829 MB	48 MB	POP3S
TCP 182	706 MB	15 MB	
UDP 443	653 MB	260 MB	
UDP 53	525 MB	331 MB	DNS

Table 3: The top 10 TCP and UDP ports sorted by the total number of received bytes.

not entirely unexpected and opens the possibility of employing caching, compression, and various other data reduction techniques as we will explore in Section 5.

Not unexpectedly, users seem to utilize our service for *mostly entertainment purposes.* Table 4 lists the top-15 websites by the volume of traffic that our users generate from each destination (most of these destinations are also our top 15 when sorted by the number of flows). For comparison we have also included the destination's rank as calculated by comScore (Nov 2011, US Top 50 list), a market research firm that monitors participants' network traffic and Alexa.com (Nov 2011, US rank) which ranks based on data collected from users that install its browser toolbar. Apple provides the highest amount of downlink traffic from a single domain, much of which includes their iTunes service. There are also a lot of other traffic from Apple that are specific to various mobile applications on different devices. There are further streaming media content flows from YouTube, Pandora, Megavideo, Netflix, and other such sites. In the uplink side, the highest amount of content flows into Facebook, indicating many of our users are uploading personal content utilizing our service.

Further, the top destinations exhibit significant differences between the Coach buses (plying inter-state routes) and the

Destination	Bytes Received	Bytes Sent	Alexa Rank	comScore Rank
apple.com	15 GB	623 MB	21	12
youtube.com	9.5 GB	361 MB	3	NA
googlevideo.com	7.7 GB	224 MB	NA	NA
limelight networks.com	6.6 GB	238 MB	185893	NA
facebook.com	6.2 GB	1.1 GB	2	4
pandora.com	4.3 GB	144 MB	68	NA
edgesuite.net	3.7 GB	112 MB	2547	NA
google.com	3.0 GB	419 MB	1	1
megavideo.com	2.8 GB	85 MB	471	NA
netflix.com	1.6 GB	79 MB	23	41
mzstatic.com	970 MB	69 MB	30078	NA
windows update.com	688 MB	25 MB	NA	NA
symatec liveupdate.com	657 MB	20 MB	NA	NA
tumblr.com	641 MB	39 MB	25	NA
yahoo.com	599 MB	92 MB	4	2

Table 4: The top 15 web destinations sorted by the total number of received bytes. For comparison we provide comScore's Nov 2011 Top 50 rankings as well as the Nov 2011 Alexa.com page rank.

Metro buses (plying city routes in Madison, WI). For instance, some of the top destinations in Metro buses, that do not figure in the Coach buses, include the National Public Radio (npr.org), the Onion (an American satire news publication with strong ties to Madison, WI), and cityofmadison.com, all of which have relevant local context in Madison, WI.

5. DATA REDUCTION TECHNIQUES

A significant cost of a vehicular Internet service stem from the monthly charges that need to be paid for cellular data plans. In our current deployment, we limit each bus to using two cellular modems to limit cost. Given the interest of cellular operators to impose various forms of bandwidth caps on cellular usage, it is critical to design efficient mechanisms that reduce the volume of uplink and downlink traffic in the system.

Given the dominance of HTTP traffic (Table 3), it is natural to expect that various forms of caching and compression techniques to be greatly beneficial. In this section, to explore these options, and also evaluate the usefulness of other data redundancy elimination techniques within packets, as proposed in recent literature [2].

5.1 Compression

When an application requests a web page it often specifies that it is capable of decompressing zipped content files. When applications make requests it is often to the benefit of the server to compress the content to both reduce bandwidth consumption as well as improve the download speed for the end user. However, not all servers compress all the content even when applications request compressed content. In such a situation a compression proxy could compress the content before it is forwarded to the end user.

For our WiRover system a compression proxy could be added to the controller node to compress the downlink HTTP content before it is transferred over the cellular downlinks to achieve potential savings. To analyze how beneficial a compression proxy would be to the WiRover system we collected detailed packet traces at the gateways of two of our Coach buses. The traces were collected over multiple days and total over 10 hours of traffic. The relatively small amount of this collection stems from the difficulty of offloading traces from the gateways. We had to do this in brief periods of time when we had access to the corresponding buses (at a stop near our lab), or over the expensive cellular links. (Note that trace collection for compression analysis at the controller would have been simple but would not have been very helpful, due to traffic being modified via NAT at the gateway, and we would lose end client-specific information if logged at the controller.)

What was compressed: We then used these traces to analyze HTTP flows and found that 5,839 out of the 17,959 HTTP flows (corresponding to 188 MB of content) were already compressed by content servers. The compressed 5,839 flows contributed nearly 18 MB of content out of these 188 MB. When we uncompressed the compressed content, we realized that the original content volume was 79 MB which has been compressed down to 18 MB. Overall, the Internet-based servers had served 249 MB of HTTP content which was compressed down selectively to 188 MB — *a total savings of about 24.5%.*

We found that most of the compressed HTTP flows contained HTML, CSS, XML, javascript, or other text based content. Interestingly, we found that some servers had compressed images which when uncompressed were actually smaller than the compressed data size. This is due to the fact that compressed files must include a dictionary of replacement values that are used during the decoding process. We observed that 564 flows displayed such behaviors but accounted only about 13 KB of waste which is dwarfed by the (79 - 18 =) 61 MB that were saved thanks to content server compression.

What was not compressed: We also found that there were 10,350 flows which were served uncompressed when the client devices had advertised compression support.

To better understand the potential gains that compression could have provided in these situation we took each HTTP payload and ran the standard gzip compression, at the highest compression levels, and measured the file sizes before and after. The uncompressed flows accounted for a total of 170 MB of data and compression would have reduced that number to about 153 MB, a reduction of over 17 MB, or about 10%. Given that the highest monthly data consumption on the buses today routinely exceed 10 GB, *we could anticipate a wasted opportunity of eliminating of more than 1 GB of data consumption at this rate.*

For these uncompressed flows we analyze the content type for each flow and find that the majority of uncompressed flows contain images, audio files, or video files. Such content is often viewed as incompressible since the data is already stored in an optimized format, however, with enough volume even savings of only a few bytes per flow can add up to significant savings. Table 5 shows the top 10 uncompressed content types sorted by potential savings. The "text/plain" content type was only observed from netflix.com related servers and the "application/octet-stream" was only observed from pandora.com servers. Both of these content types display high volumes of traffic with relatively low compression ratios compared to text based content such as "text/html" or "application/atom+xml" both of which have smaller total bytes but have higher compression ratios.

Content Type	Total Bytes	Potential Savings	Total Flows
image/jpeg	23 MB	4.1 MB	463
image/png	12 MB	2.5 MB	351
text/plain	23 MB	2.2 MB	164
application/ octet-stream	39 MB	1.4 MB	29
application/ javascript	1.7 MB	1.4 MB	8
video/mp4	9.6 MB	720 KB	27
application/ x-rhapsodycontentaac	23 MB	539 KB	8
text/html	533 KB	401 KB	3
application/atom+xml	437 KB	395 KB	2
video/x-flv	15 MB	388 KB	4

Table 5: The top 10 HTTP content types sorted by the number of bytes that would have been saved if the content had been compressed using gzip.

Period	1	2	3	4	5
Total (MB)	47	57	56	59	59
Savings (MB)	8.5	8.3	5.0	16	6.1
Savings (%)	18.1	14.6	8.9	27.1	10.3
Period	**6**	**7**	**8**	**9**	**10**
Total (MB)	50	57	55	58	56
Savings (MB)	11	13	12	6.8	18
Savings (%)	22.0	22.8	21.8	11.7	32.1

Table 6: Savings in a 10-hour trace using Packet Caches between the controller and the gateway.

5.2 Packet-level redundancy elimination

An interesting optimization that could be added into the WiRover system would be to eliminate redundancies at the packet level using techniques such as Packet Caches [2]. That proposed technique creates fingerprint hashes of matching substrings of packet payloads and stores them at the receive side. If the same substrings are found in future packets (quickly detected by comparing hashes), then the matching substring can be eliminated from the data packet and be replaced by an appropriate identifier. One advantage that this method has over HTTP compression and caching is that it operates over all traffic flows and is not limited to HTTP traffic. Further, given that the vast majority of our traffic is HTTP based this method is also able to compress the text based HTTP *headers* that are found in each HTTP flow. Note, that server-based content compression techniques that operate on HTTP flows do not compress HTTP headers, which can leave out significant opportunities when the HTTP payloads are small (especially with many occurrences of 1 pixel by 1 pixel images in content these days).

Table 6 shows the possible compression achievable by using Packet Caches deployed across the controller and gateway pair for the 10-hour trace, partitioned into approxi-

Type	Hit Count	Total Bytes
image/jpeg	6408	77 MB
application/javascript	4028	38 MB
image/png	5302	30 MB
application/zip	2532	26 MB
text/css	2522	21 MB

Table 7: The top 5 HTTP content types sorted by the total number of bytes served locally by the web cache.

Failure Type	Count
Hardware failures	4
No connectivity on boot-up	403
Individual cellular connectivity failure	3,624

Table 8: The number of failures by type.

mately 1 hour periods. The achievable savings range between 8.9% and 32.1% for the most part.

5.3 Caching

To further reduce resource consumption on the wireless back-haul networks a web cache could be installed on each gateway node to serve cached content locally. To test the effectiveness of web caching on our WiRover system we ran a squid proxy (using the default configuration) for a little more than two months (July 28, 2011 to Oct. 10, 2011) on a single Coach bus. Over this period of time 36,270 cache hits were observed out of 265,867 total events. The total volume of traffic that was served locally due to the web cache amounted to over 286 MB out of a total of 7.8 GB.

Table 7, shows the top 5 HTTP content when sorted by the number of bytes served locally. As is seen the majority of gains come from image files. For these image files we find that the 6408 hits correspond to 4395 unique URLs. Of these unique URLs only 981 recorded only a single hit while 428 and 181 recorded 2 and 3 hits respectively. The recording the most hits (32 in total) was a logo icon from www.facebook.com.

Caching Considerations: Without knowledge of the client device's IP address it is difficult to determine whether these cache hits are the result of a single user downloading the same content multiple times or if there are multiple users accessing the same content. However, the WiRover system would benefit in either case.

It is further unclear to what extent browser caching benefits a systems such as ours. Since we do not have access to client devices this question is out of the scope of this particular piece of work.

6. FAILURE ANALYSIS

To ensure our system is providing a reliable service it is imperative that we monitor system failures and repair them quickly. System failures can range from uncontrollable events such as a lack of cellular network coverage to hardware specific failures that require replacing the gateway unit on a bus. Some of these failures can be extremely difficult to detect and diagnose and have put in place significant checks and balances to quickly detect and address them (see Section 2).

	Total			Avg Link
Provider	Failures	Metro	Coach	Uptime (min)
NetA	2,807	270	2,537	147.6
NetB	817	301	516	457.0
Total	3,624	571	3,053	

Table 9: The number of individual cellular connectivity failures by provider.

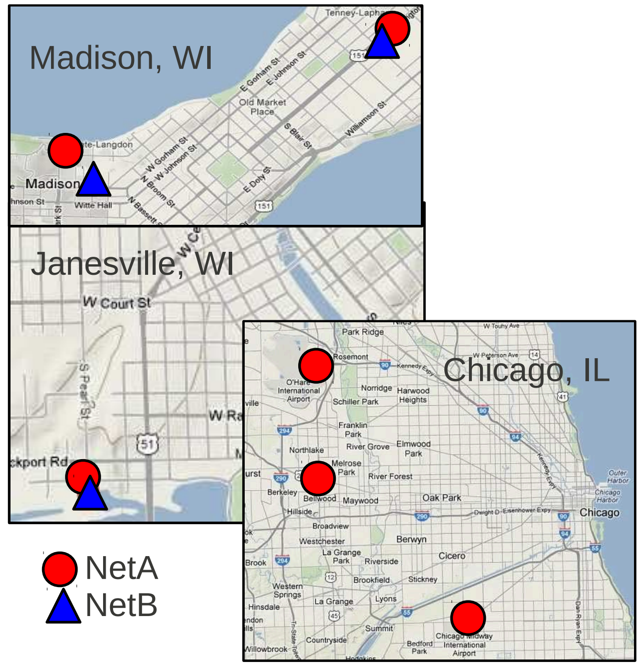

Figure 14: Locations with the highest density of link failures for each network.

Table 8 groups the various system failures we have encountered over the past 18 months. There were four occurrences of hardware related failures. Our first hardware failure was the result of an improper vehicle power supply (Section 3). The other three failures have been caused by a combination of hard disk failures and system clock skew. However, we routinely observe transient failure scenarios that are in some way related to connectivity issues. There were 403 recorded instances when a gateway booted-up, but none of the cellular interfaces were able to lock onto their corresponding cellular base stations. Our system is configured to re-boot itself after a 3-minute period elapses in this state. We have seen many occurrences of this event primarily in the bus garages, where various forms of RF shielding from the environment may cause this sort of disruption. In addition, there were 3,624 instances where one of the functioning cellular interfaces suddenly lost connectivity (as reported by the cellular interface card).

6.1 Causes of Link Failures

Table 9 shows the number of individual failures experiences by the different cellular interfaces. The table also shows a comparison of the number of link failures observed on each type of bus as the Metro buses only cover an urban area while the Coach buses also travel in rural areas.

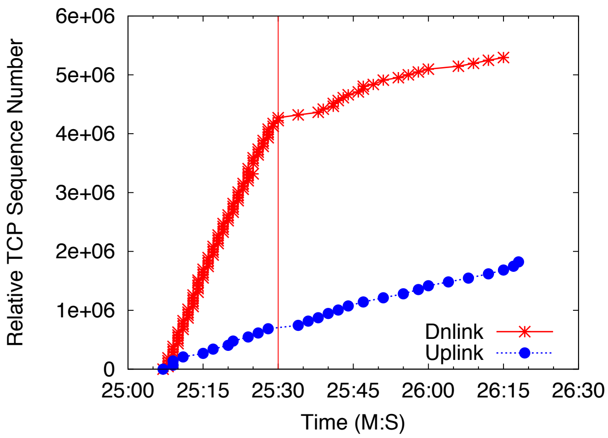

Figure 15: TCP sequence numbers for the downlink and uplink iperf streams. One of the cellular modems was unplugged at the time indicated by the vertical line.

By clustering link failures by their GPS location we are able to determine locations that have a history of causing connectivity failures in individual cellular interfaces. Figure 14, provides the locations where the most failures occur in a 1 km radius area. Not surprisingly, 2 of the 3 locations for which NetB is represented are the locations of the garages for both Metro buses and Coach buses. Historical data such as this could then be coupled with real-time measurements to improve our understanding of what causes each such failure.

6.2 Link Failure Prediction

We have observed that cellular interface cards actually take a long time to report that it has lost connectivity to the networks. Sometimes it may be as much as 10-15 seconds after the link starts to lose packets sent along it. In fact, 10-15 seconds prior to the interface reporting a failure, we observe an increased occurrence of packet losses from that network. This observation has been routinely observed in the link failure scenarios described above. It implies that one can use an increased occurrence of packet losses to predict the onset of a connectivity loss for an interface relatively easily.

An alternative to this design could be to use the received signal strength indication (RSSI) metric to estimate the quality of cellular connectivity and likely onset of a disconnection period as proposed in the MAR system. However, the APIs in our cellular code division multiple access (CDMA) modems cannot be used to query RSSI values while the modem has an established data connection to a cellular network. Hence, we believe a loss rate estimation technique might be a fast and efficient way to predict onset of link failures and to initiate flow migrations.

6.3 Flow Migration

Often times, link failures are caused by factors that are not controllable by our WiRover system. For example, if the bus enters an area with little to no signal strength then a link failure is likely to occur. Our goal is to ensure that our system is capable of mitigating such link failures. This is an important goal for user retention.

Cellular link failures can be frustrating for users in the vehicular environment but often times they cannot be avoided. For our WiRover system to provide users with a superior experience our system must then mitigate the effect of link failures. One mechanism the WiRover system uses to mitigate link failures is to migrate network flows from the failed network to an active network. However, it is important to understand the effect link migrations have on network traffic.

To best understand the effects of link migrations we show an example of how such flow migration operates over our WiRover system. This was a bidirectional TCP flow operating between a client device and an Internet-based server. We plot the TCP sequence numbers for both the uplink and downlink parts of the flow in Figure 15. The vertical line in the figure indicates the time at which connectivity to one of the cellular networks was lost. As can be seen, the downlink flow experiences some packet loss, and a brief stall in performance, but quickly migrates to the other interface and resumes to send packets on the new cellular link. The uplink direction, in this example, was not originally assigned to the failed cellular network, and hence did not get affected.

7. RELATED WORK

Providing Internet connectivity into a vehicle is not a new idea. Early work studied the challenges faced when attempting to connect to a wireless basestation from a vehicle using WiFi based technologies [18, 11, 13]. Multiple other works have attempted to connect to open WiFi access points to provide a reliable Internet connect into a vehicle [7, 15, 9, 12, 16, 8]. In particular Cabernet [9], proposed multiple optimizations to the process of connecting to a WiFi AP which in turned increased the amount of time they were able to maintain a connection while driving by.

The MAR system [21] avoided many of the challenges of vehicular Internet connectivity by using cellular technologies instead of WiFi based technologies. The MAR system implemented a multihomed designed which used multiple 2.5G and 3G cellular links from different cellular providers. Such a design allowed them to exploit network diversity and aggregate bandwidth to improve the reliability and performance of their system. The authors proposed multiple traffic scheduling mechanisms and implemented a “per-TCP connection state scheduler.”

The PluriBus project [14] based their design from the MAR system but implemented their system with a combination of 3G cellular and WiMAX modems. The authors observed high loss rates on their WiMAX links and developed evolutionary codes to mitigate those losses.

A recent project (Wiffler) [4] explored the feasibility of offloading cellular traffic to WiFi networks. Other projects [6, 3, 5] have studied the performance of various application classes such as web search and interactive application from vehicular settings.

Unlike all such prior work, our effort focused on providing a continuous and reliable Internet connectivity service to these public transit passengers. This paper shares our experience in running such a service which continues to challenge us far beyond than rolling out and deploying an experimental testbed.

8. OPEN RESEARCH CHALLENGES

Our toils over the past 18 months have exposed challenges that we had not anticipated based on prior work in vehicular environments.

Management: We have found that managing a vehicular deployment is particularly challenging. The ability to track and monitor the health of mobile devices is challenging in itself. However, such a deployment generates troves of data that must be collected and analyzed which requires automated mechanisms to collect both raw usage data and user feedback data.

Traffic Optimizations: Our desire to expand our deployment is hampered by the financial burden of our cellular based connectivity. Recent works have suggested offloading data to less expensive technologies [4] but in rural areas other technologies may not be readily available. As such other traffic optimizations are needed to help reduce the network load on cellular back-haul links.

Link Failure Prediction: We have found that wireless link failures are extremely common in the vehicular environment. The ability to predict link failures could greatly improve the reliability of the system.

9. CONCLUSION

Deploying a system in the wild has provided many challenges over the past year. However, with the large number of users that have connected to our system we are continually working to improve the performance of the system so that we can support more and more users. We hope that in the long-term our WiRoversystem will provide valuable insight into the vehicular networking environment. However, in the short-term we are happy to provide a service to the community while we further our research interests.

Overall, our more than 18 months of operation has provided us with various insights on how performance in these vehicular Internet access systems can be tuned. While many other vehicular Internet services do exist today, little is known about their actual performance or potential pitfalls. In fact, when talking to one of our bus operators (Van Galder Bus) who have had experiences with a few other such systems at different times, *we were told that many existing systems have very unpredictable performance.* Performance can be quite spotty, and failures are unexpected, with the vendor being totally unaware of the duration, the period, and most importantly, the cause of such failure. While resetting the node on the vehicle often fixes the problem, without a systematic study, such problems are bound to recur. We hope that a study such as ours can help shed light on some of the issues one can expect in such vehicular systems.

Both our partner bus operators have been fairly satisfied with the performance of the WiRover system so far. Our users have voted with their feet (or their bytes), as we continue to see a growing number of repeat users. Most interestingly, there were a large number of smart phone users, who in spite of their existing phone data plans, form the highest user population in our system. Our study has provided us (and hopefully, others) with many insights on how to improve this system even further, and we hope to implement them in the months to come.

10. ACKNOWLEDGMENTS

We would like to thank Rob Madden for his significant implementation contributions on an early version of the system as well as Wes Miller and Sam Roth for efforts to create our management tools. We thank the Madison Metro Transit and the Van Galder Bus Company for partnering with our research efforts and for their feedback and suggestions. All authors are supported in part by the following grants of the US National Science Foundation: CNS-1040648, CNS-0916955, CNS-0855201, CNS-0747177, CNS-1064944, and CNS-1059306.

11. REFERENCES

[1] Amtrak to take free onboard wifi nationwide. `http://arstechnica.com/tech-policy/news/2010/06/amtrak-to-take-free-onboard-wifi-nationwide.ars`.

[2] Ashok Anand, Archit Gupta, Aditya Akella, Srinivasan Seshan, and Scott Shenker. Packet caches on routers: the implications of universal redundant traffic elimination. In *SIGCOMM*, 2008.

[3] Aruna Balasubramanian, Brian Neil Levine, and Arun Venkataramani. Enhancing interactive web applications in hybrid networks. In *MobiCom*, 2008.

[4] Aruna Balasubramanian, Ratul Mahajan, and Arun Venkataramani. Augmenting mobile 3G using WiFi. In *MobiSys*, 2010.

[5] Aruna Balasubramanian, Ratul Mahajan, Arun Venkataramani, Brian Neil Levine, and John Zahorjan. Interactive WiFi connectivity for moving vehicles. In *SIGCOMM*, 2008.

[6] Aruna Balasubramanian, Yun Zhou, W. Bruce Croft, Brian Neil Levine, and Aruna Venkataramani. Web search from a bus. In *CHANTS*, 2007.

[7] Vladimir Bychkovsky, Bret Hull, Allen Miu, Hari Balakrishnan, and Samuel Madden. A measurement study of vehicular Internet access using in situ Wi-Fi networks. In *Mobicom*, 2006.

[8] Pralhad Deshpande, Anand Kashyap, Chul Sung, and Samir R. Das. Predictive methods for improved vehicular wifi access. In *MobiSys*, 2009.

[9] Jakob Eriksson, Hari Balakrishnan, and Samuel Madden. Cabernet: vehicular content delivery using wifi. In *MobiCom*, 2008.

[10] Jakob Eriksson, Lewis Girod, Bret Hull, Ryan Newton, Samuel Madden, and Hari Balakrishnan. The pothole patrol: using a mobile sensor network for road surface monitoring. In *MobiSys*, 2008.

[11] Richard Gass, James Scott, and Christophe Diot. Measurements of in-motion 802.11 networking. In *WMCSA*, 2006.

[12] Anastasios Giannoulis, Marco Fiore, and Edward W. Knightly. Supporting vehicular mobility in urban multi-hop wireless networks. In *MobiSys*, 2008.

[13] David Hadaller, Srinivasan Keshav, Tim Brecht, and Shubham Agarwal. Vehicular opportunistic communication under the microscope. In *MobiSys*, 2007.

[14] Ratul Mahajan, Jitendra Padhye, Sharad Agarwal, and Brian Zill. E PluriBus Unum: High performance connectivity on buses: Technical Report: MSR-TR-2008-147, October 2008. `http://research.microsoft.com/apps/pubs/default.aspx?id=70650`.

[15] Ratul Mahajan, John Zahorjan, and Brian Zill. Understanding wifi-based connectivity from moving vehicles. In *IMC*, 2007.

[16] Anthony J. Nicholson and Brian D. Noble. Breadcrumbs: forecasting mobile connectivity. In *MobiCom*, 2008.

[17] Justin Ormont, Jordan Walker, Suman Banerjee, Ashwin Sridharan, Mukund Seshadri, and Sridhar Machiraju. A city-wide vehicular infrastructure for wide-area wireless experimentation. In *Mobile Computing and Networking*, 2008.

[18] J. Ott and D. Kutscher. Drive-thru internet: IEEE 802.11b for "automobile" users. In *INFOCOM*, 2004.

[19] Feng Qian, Zhaoguang Wang, Alexandre Gerber, Zhuoqing Morley Mao, Subhabrata Sen, and Oliver Spatscheck. Characterizing radio resource allocation for 3g networks. In *IMC*, 2010.

[20] Asfandyar Qureshi and John Guttag. Horde: separating network striping policy from mechanism. In *MobiSys*, 2005.

[21] Pablo Rodriguez, Rajiv Chakravorty, Julian Chesterfield, Ian Pratt, and Suman Banerjee. Mar: A commuter router infrastructure for the mobile Internet. In *MobiSys*, 2004.

[22] Sayandeep Sen, Jongwon Yoon, Joshua Hare, Justin Ormont, and Suman Banerjee. Can they hear me now?: A case for a client-assisted approach to monitoring wide-area wireless networks. In *IMC*, 2011.

[23] Hamed Soroush, Nilanjan Banerjee, Aruna Balasubramanian, Mark D. Corner, Brial Neil Levine, and Brian Lynn. Dome: A diverse outdoor mobile testbed. In *HotPlanet*, 2009.

[24] The New York Times. Google's buses help its workers beat the rush. `http://www.nytimes.com/2007/03/10/technology/10google.html`.

[25] USA Today. More cities offer wi-fi on buses. `http://www.usatoday.com/tech/wireless/2008-04-10-wifi\_N.htm`.

[26] Xiaolan Zhang, Jim Kurose, Brian Neil Levine, Don Towsley, and Honggang Zhang. Study of a bus-based disruption-tolerant network: mobility modeling and impact on routing. In *MobiCom*, 2007.