Tracking Buses to Track Smartphone Users

Ryan Eckenrod
University of Maryland
eckenrod@umd.edu

ABSTRACT

The GPS location of a mobile device often corresponds to its ownerś location and is thus restricted to mobile apps which the user explicitly grants access permission. Meanwhile, real-time vehicle tracking for public transportation systems is increasingly available to the public in cities and many large universities. Using one such vehicle location API, we demonstrate a new side-channel localization attack by which a victim can be precisely geolocated if nearby buses from a public transit system can be detected by a malicious app. To evaluate this attack, we conduct 60 experiments of 200 localization attempts each across two regions (a large city and a university). We find that within half an hour, depending upon the capabilities of the bus detection method used, up to 39% of victims near a bus route can be located to within 20m of their exact location in either region. Using a simple early stopping heuristic, we also find that the time to detect some victims can be reduced to approximately 16 minutes with little degradation in localization accuracy.

1 INTRODUCTION

A person's location at any given time is often regarded as private information that should not be accessible by arbitrary third parties without permission. This general rule is enforced in both Android and iOS mobile devices, which require users to explicitly grant location information access to any downloaded mobile apps desiring either the device's GPS location or a coarse location estimate based on cellular or WiFi connectivity. They additionally notify users constantly when their location information is being accessed by such applications. Therefore, a means of undetectably and accurately determining a user's location via their mobile device without explicitly requesting it is a dangerous violation of user privacy. Multiple side-channel attacks have been presented for this purpose, using seemingly innocuous permissions such as device battery life or accelerometer readings to locate a victim user [10, 14].

In this work, we propose a novel approach for covert user location inference by detecting a user's proximity to public transit vehicles in an outdoor environment. Real-time vehicle tracking for public transportation systems has increasingly become available to the public through APIs such as NextBus or MTA Real-Time Data Feeds [1, 2]. Additionally, some transportation systems have deployed WiFi access points on public buses [9, 18]. Based on these developments, we advance that a user in a city sized area can be precisely geolocated by a malicious app on his or her mobile phone given permission to access a device sensor that could be used to detect nearby buses, such as the device's microphone or WiFi state. Indeed, buses are louder than normal traffic, and recent works classifying cityscape sounds have demonstrated that the sounds of buses, trains, or cars can be distinguished relatively easily by audio classifiers [7, 8].

To test this theory, we design a Bayesian localization method that combines "bus" or "no bus" observations extracted covertly from a victim user's mobile device with downloadable real-time bus locations to determine the most likely location of the victim in a city-sized area. We pair this localization method with simulated nearby bus detectors (classifiers) of varying detection range, precision, and sensitivity to locate victims within a large university environment and within the city of San Francisco, and we examine how the quality of the bus classifier affects localization success in both regions. We find that after using this method for half an hour, between 16% and 39% of victims near a bus route within UMD can be located to within 20m of their exact location depending upon the bus detection method used, and 28.5% to 69.5% can be located to within 80m of their exact location. For the larger area of San Francisco, 10% to 30.7% of victims can be located to within 20m of the correct location, while 12% to 54.2% can be located to within 80m. Additionally, we find that applying a simple early stopping heuristic to the method can often provide victim location predictions in 15 to 16 minutes with minimal impact on prediction accuracies.

2 RELATED WORK

Our work bridges the gap between legitimately useful context-aware localization applications and malicious side channel attacks which attempt to use non-GPS sensors to locate users. One example of a context-aware system is SurroundSense [5], which integrates ambient noise, accelerometer data, light, and image measurements to identify which room a user is in when indoors. Ambient noise is also used for indoor localization in a number of other studies to perform Time Difference of Arrival (TDOA) localization [6, 11, 12]. However, these techniques have been sparingly applied to city-sized outdoor environments and rely on physically deploying multiple devices which produce sounds at predetermined times in the environment to be effective. Other studies locate users based on the variance or change of signal strength from nearby WiFi access points or GSM cell towers [4, 13, 15, 17].

Because many devices have defenses or permission systems in place to restrict access to GPS or signal strength measures without consent from the user, covertly geolocating a user is a challenging problem. Still, there are many creative and effective solutions. Zhou et. al explore a wide variety of attacks for locating users with Android devices without acquiring any privileged device permissions [19]. In one interesting example, the authors are able to determine the driving route of a user based simply on whether the `isMusicPlaying` variable is enabled or disabled due to instruction dictation from the Google Maps app. Another attack tool, "Powerspy", combines power usage fingerprints for an area and aligns them with extracted aggregate power usage from a victim's phone to determine which route the victim is on [14]. A similar method is employed by the ACComplice attack, which uses accelerometer data correlated with public map data to infer a user's outdoor route

[10]. At the time of this writing, no special permissions are needed to access the accelerometer; the malicious app only needs to obtain network privileges in order to send the accelerometer data to the attacker.

Side-channel localization attacks rely on the assumption that a victim user has already downloaded a malicious app to their mobile device. Researchers have estimated the proportion of malicious Android apps to be as high as 0.5% [20], which is non-negligible considering the millions of potential victims with Android phones. An example of an malicious app which could provide the sensor readings necessary for the previously mentioned attacks is Soundcomber [16]. This malware app acts as a trojan horse which selectively detects sound to identify PIN and phone numbers based on the tones they produce when pressed. Results of these studies justify the assumption that users may download malicious applications which would allow the previously mentioned (and currently proposed) attacks to be performed.

Our work is most similar to the side-channel attack studies, but also incorporates some of the modeling techniques from the context-aware systems. The main limitation of prior side-channel localization attacks is the need for a "training phase" in which the attacker manually collects measurements at different locations in the target region. For example, Powerspy [14] requires manually measured power usage data to be collected by the attacker for several routes in an area. The only training phase necessary for our project is an empirical test of the bus detector, which can be performed once in virtually any location and will then work in any target region with a public bus system providing live vehicle location APIs.

3 LOCALIZATION VIA NEARBY BUSES

3.1 Threat Model

We first assume that the victim downloads a malicious app created by the attacker to their mobile device. The app has no permission to access the device's GPS location or any other information that leads to an easy location estimate such as nearby cell towers. The app only requires network access to communicate with the attacker through a remote server and access to a device permission which could be used to detect a nearby bus. Some example permissions we believe could be used include microphone access to detect the sound of buses or WiFi state access to detect WiFi enabled buses by their access point SSIDs. The first goal for the attacker during a localization attempt is to use the bus detecting permission as a classifier which determines if the victim is near a bus or not at fixed intervals over time. To do so, the app runs the classifier continuously in the background of the device, relaying whether a bus is detected or not to the attacker periodically. We assume that the victim is stationary for the duration of the localization attempt, but we believe our method could be extended to locate a mobile victim as well.

We next assume that the attacker has access to a public transit vehicle tracking API for the area in which the victim is believed to be. From the remote server, this API can be used to query the current locations of the transit system's buses in real time. Using "bus" or "no bus" outputs from the app's nearby bus classifier along with the API's live bus location information over time, the attacker attempts to predict the victim's location in the area. The attack is thus limited to areas with transit systems that make vehicle locations publicly available, but such services are often available in large to medium sized cities and universities [3]. The attack is also limited in that victims who are not near bus routes cannot be located; a victim whose phone never hears a bus during the localization attempt could be in any location not near bus routes with equal probability. On the other hand, we show that a victim who hears even a single bus at one time interval may be correlated to a specific position over time.

3.2 Localization Methodology

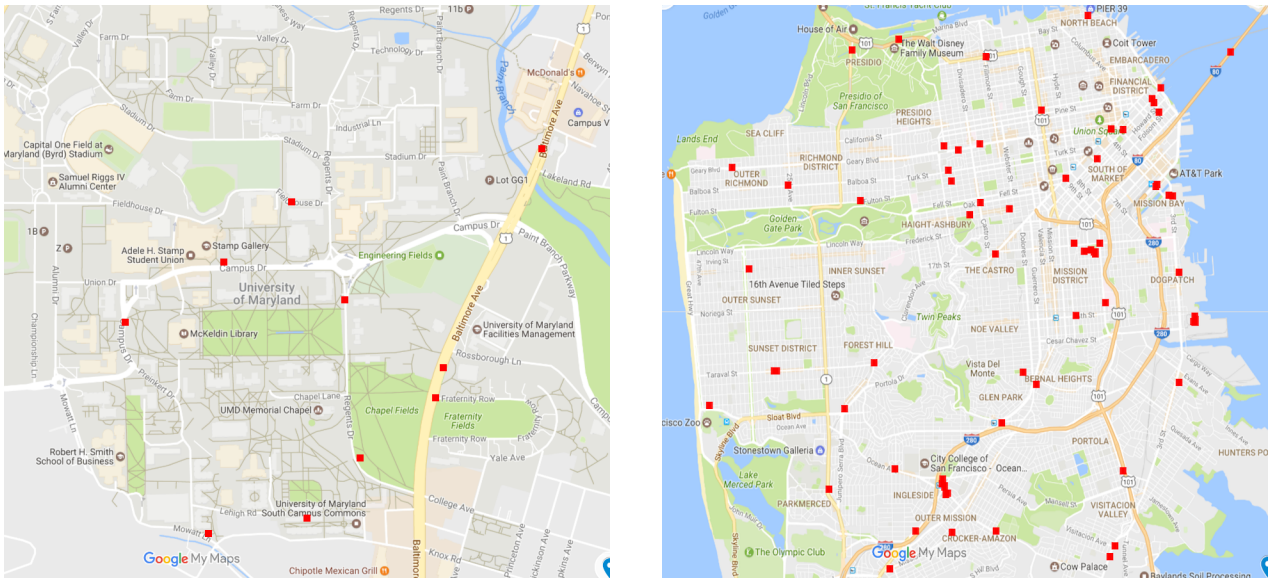
Given this threat model, we reduce locating the victim to identifying their most likely position in an area given a series of "bus" or "no bus" observations from the app and live bus location information. We first model the city sized area in which the victim resides as a grid of equally sized square cells and create a distribution of probabilities (beliefs) that the victim is in each cell which is initially uniform. We then approach the problem as an instance of bayesian modeling where we seek to find the position of the victim X at time t to maximize $P(X|b_1 \ldots b_t)$ where $b_1 \ldots b_t$ is the sequence of bus observations of the form "bus" or "no bus". Using Bayes' rule, this can be written as $P(X|b_1 \ldots b_t) = \frac{P(b_t|X)P(X|b_1 \ldots b_{t-1})}{P(b_1 \ldots b_t)}$, and because $P(b_1 \ldots b_t)$ does not affect the maximization, we can write $P(X|b_1 \ldots b_t) = cP(b_t|X)P(X|b_1 \ldots b_{t-1})$. This gives us the basic Bayesian update rule for new evidence: for each periodic app classification, for every cell in our distribution, we simply update the probability that the victim is in the cell by multiplying the current probability that the victim is in the cell by the probability of the evidence given the distance from the cell to the nearest bus (as is revealed by the public bus tracking APIs). After each step, we normalize the cell probabilities so that they sum to 1. After a sufficient number of app readings (defined by either a time limit or a when maximum probability threshold is reached), the cell with the highest probability is predicted as the victim's location[1].

The question remains how to calculate the probability of observing a bus from a given cell in order to perform this update. In practice, this depends on the quality of the 'bus", "no bus" classifier, which is likely a function of the distance to the closest bus. Rather than creating and testing a specific classifier, we simulate classifiers of varying performance to determine our Bayesian localization method's usefulness when paired with different classifiers.

4 EXPERIMENTS

We next describe how we test our localization method. Instead of implementing a malicious app and attempting to localize test victims in real time, our experiments simulate these victims and attempt to localize them using saved bus location data in order to run the experiments more quickly. We answer the following questions with our experiments:

[1]While this simple update method was ultimately used, we also implemented a Monte Carlo particle filter which has a runtime linear in the number of particles instead of linear in the number of cells. However, the "direct" Bayesian model described above did not take very long to compute even for San Francisco (a few seconds per iteration on commodity hardware), so we elected to use this simple method instead.



(a) University of Maryland

(b) San Francisco

Figure 1: These maps correspond to the two areas in which we attempt to localize simulated victims, The University of Maryland and San Francisco. Red squares on these maps indicate the GPS locations of buses in each area at two uniformly selected times from the recorded range.

- Given different 'bus", "no bus" classifiers, how often can our method locate a victim near bus routes in a given area, and with what accuracy?
- How does varying the bus detection range, false negative rates, and false positive rates of classifiers affect the method's localization accuracy?
- Can victims be located quickly without hurting localization accuracy?

4.1 Regions Tested

We seek to measure the localization accuracy of our method in two different environments: a large university with a modest local bus system and a large city with a vast bus fleet. We therefore choose to run our experiments over the regions of the University of Maryland, College Park (UMD) and San Francisco, California, both of which have bus systems using the NextBus API [2]. For each region, we first recorded the location (latitude and longitude) of every bus every ten seconds between Friday, 6:00am EST 04/21/17 to Saturday, 3:00am EST 04/22/17 by querying the NextBUS API. Bus locations between these times thus represent a typical weekday bus schedule which we use as a model for weekday bus activity. We also used the API to get the route information for each region: granular sets of GPS locations which buses on each route drive over. Using this route information, we define a 2D grid representing each region. Each grid is comprised of square cells 20 meters in length. These grids initially contain all locations that the buses may drive over (derived from the bounds of the route information), but we scale them down to exclude portions of each region where buses rarely travel. The final grid for the University of Maryland represents a 2.43km^2 (0.93 mi^2) area, while the grid for San Francisco represents 153.38km^2 (59.22 mi^2). The bounds of these grids are displayed in Figure 1.

Max Detection Distance	(False Positive, False Negative) Rates
[20, 40 ,60]	[(0,0)]
[20, 40 ,60]	[(0.01, 0), (0.05, 0), (0.1, 0)]
[20, 40 ,60]	[(0, 0.05), (0, 0.1), (0, 0.2)]
[20, 40 ,60]	[(0.01, 0.05), (0.05, 0.1), (0.1, 0.2)]

Table 1: Values of classifier attributes which are varied across modeled classifiers for each testing region. Max distances are crossed with false positive, false negative rate tuples to create classifiers for each experiment.

4.2 Classifier Modeling

As previously mentioned, we neither aim to create a classifier for detecting buses in this work nor aim to test such a classifier in the field as part of the localization technique. We instead model multiple classifiers, varying the max distance each one can detect a bus, its false positive rate, and its false negative rate. By doing so, we can observe the performance of our localization method when paired with classifiers of varying bus detection distance, precision, and sensitivity. Table 1 shows the values of these variables we chose to sample when modeling classifiers. For instance, a classifier with a max detection distance of 40m will fail to report hearing a bus when one is 50m away during a simulation, while a classifier with a 0.2

Region	Max Distance	Cells Covered	Total Cells	Proportion Covered
UMD	1	2347	6,080	0.386
UMD	2	3597	6,080	0.592
UMD	3	4508	6,080	0.741
San Francisco	1	21,559	383,460	0.056
San Francisco	2	45,416	383,460	0.118
San Francisco	3	72,915	383,460	0.190

Table 2: Proportion of cells sampled from for experiments based on classifier max detection distance. This proportion represents the percentage of each region's cells in which a victim could theoretically be located by this method given a bus detector with each max distance.

false negative rate will fail to detect a bus in range with probability 0.2. We ultimately test 30 classifiers for each region for a total of 60 experiments.

4.3 Localization Experiments

For each simulated classifier, we run an experiment of 200 simulated localization attempts in each region. For each simulation, we first sample a random location in the grid along a bus route to place a test victim. This location is either in a cell that a bus drives through or within 1-3 cells of such a cell depending upon the maximum distance within which the modeled classifier can detect a bus. The percentage of total cells in a region which are sampled from depedning on the max detection distance can be found in Table 2. We next choose a random time within the 21 hours of recorded bus data to start the simulation, each of which lasts for half an hour. Every consecutive ten seconds after the start time, the corresponding saved bus data is used to place buses within the grid. The victim relays whether or not a bus is detected based on the classifier used, and this observation is used to update the probability that the victim resides at each cell in the grid. At each such step, we record whether a bus is observed, the cell which the victim most likely resides in according to the method, and the corresponding probability that the victim resides in that cell. The cell with the highest probability in the grid at the end of the simulation is treated as the predicted location of the victim.

5 RESULTS AND DISCUSSION

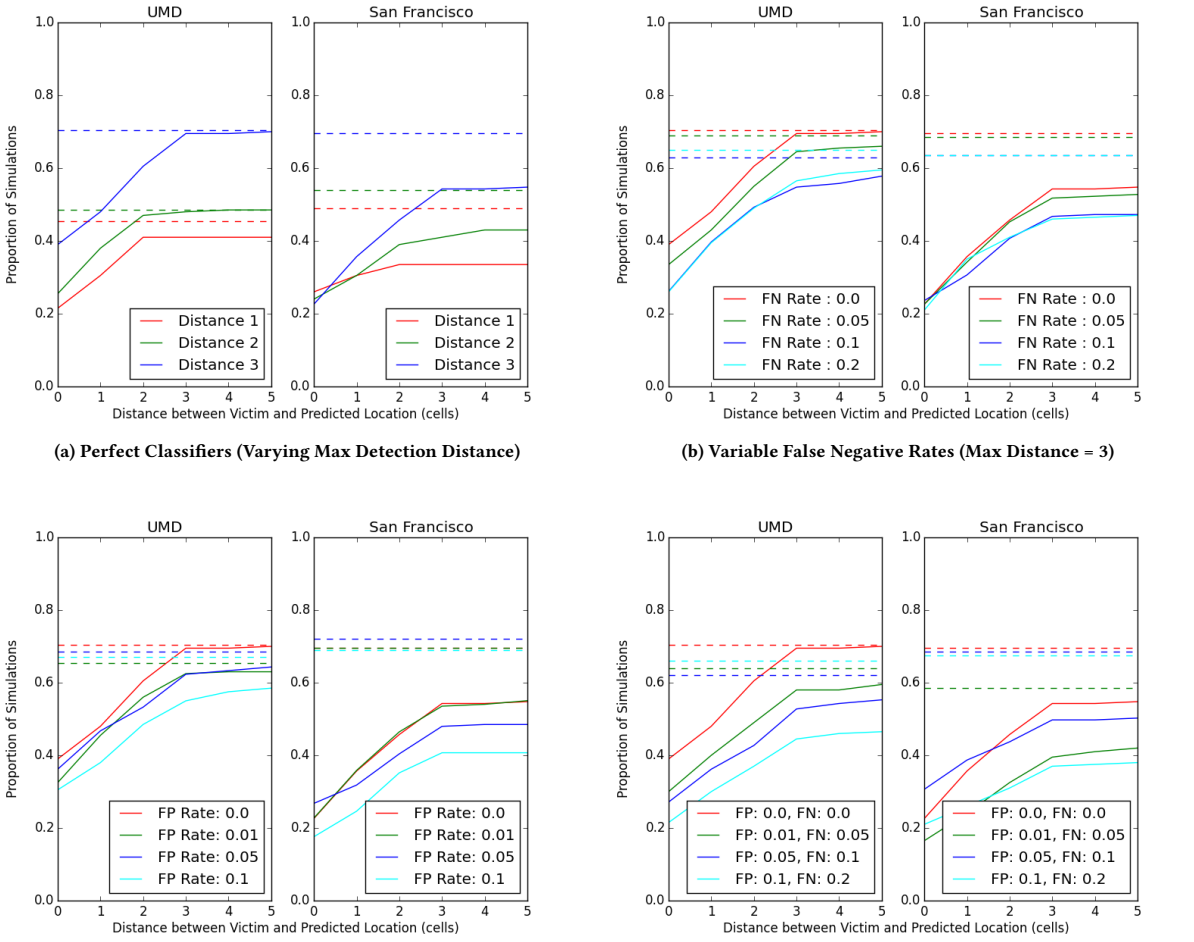
The full results of our 60 experiments can be viewed in Table 3 and Table 4 in Appendix A. We measure the success of each experiment in terms of the difference between the actual location of the victim and predicted location at the end of the half hour simulation.

5.1 Performance with Perfect Classifiers

First, experiments were conducted assuming perfect classifiers, i.e classifiers that produce no false negatives or false positives when detecting nearby buses. Though perfect bus detection may not be realistic, these experiments serve as an upper bound for how this localization method can perform. For each region, perfect classifiers with maximal bus detection ranges of 1, 2, and 3 cells away (20m, 40m, and 60m respectively) from the current cell were simulated. Figure 2(a) displays the cumulative distribution functions of these classifiers' localization accuracies for each region up to 5 cells away from the victim's real location. We do not show localization performance past this limit as we feel any localization predictions more than 5 cells away from the victim's real location, a 100m or greater error, are not useful for localization.

The dashed horizontal lines in these CDFs indicate the proportion of simulations per experiment in which at least one bus passes the victim during the simulation, indicating the total proportion of victims which the method can be expected to locate. It's important to note that in a number of simulations, the victim was never passed by a bus. These victims are impossible to locate given that many cells in each simulation are never passed by a bus, leaving many equally likely cells in which the victim may reside. In practice, any classifier which provides a negligible number of false positive detections would allow the attacker to distinguish such situations during a localization attempt and choose not to use whatever location prediction is returned by the method. In our experiments, we can observe that for a classifier with maximum detection distance of 20m or 40m outside of the current cell, roughly half of all localization attempts fail completely due to the victim never being passed by a bus for both UMD and San Francisco. When the classifier maximum detection distance is increased to 60m outside the current cell however, 70% of victims are locatable in both regions. Running longer simulations may increase these percentages as buses have more time to pass victims.

For both regions, we see that between 21% and 26% of victims can be exactly located to within a 20m x 20m cell by this localization method when it is paired with a perfect classifier of maximum detection distance 20m or 40m. 39% of UMD victims can be exactly located by a classifier with a maximum detection distance of 60m, while only 23% of San Francisco victims can be. We find that for the UMD experiments, nearly all locatable victims can be localized to within an 80m radius of their actual position as the CDF lines approach the dashed line performance limits signifying the proportion of localizable victims. For San Francisco however, the CDFs fail to approach these performance limits. This failure likely originates from the size disparity between UMD and San Francisco. For the University of Maryland simulations, the method must narrow down the victim's location within 6,080 cells to only one likely location; for San Francisco, the grid is 383,460 cells. This area is over 63 times larger than UMD, and even though San Francisco has a higher number of buses on the road, it has a much lower number of buses per square kilometer compared to UMD.



(a) Perfect Classifiers (Varying Max Detection Distance)

(b) Variable False Negative Rates (Max Distance = 3)

(c) Variable False Positive Rates (Max Distance = 3)

(d) Variable False Negative and False Positive Rates (Max Distance = 3)

Figure 2: CDFs of localization accuracy for select experiments.

5.2 The Effects of False Negatives

We next explore the effect of false negatives on the effectiveness of our methodology. We hypothesized that our localization method would be fairly resilient to false negatives because of the high rate of observations polled from the victim (every 10 seconds). It is often the case that when a bus is nearby the victim, it can be detected in multiple consecutive observations, especially by classifiers with larger detection distances. As such, even if one observation falsely reports that no bus is nearby, if following iterations correctly observe the bus, the error will be almost immediately corrected. We also believe these experiments might be representative of the performance than a classifier using nearby WiFi SSIDs to detect buses may achieve in practice. While false negatives may occur due to obstacles blocking the WiFi signal near a victim, false positive errors should not occur if bus WiFi access points have SSIDs easily distinguishable from other WiFi networks.

Classifiers with false negative rates of 0.05, 0.1, and 0.2 yet no false positives were modeled for each maximum detection distance for each region. Figure 2(b) shows the localization accuracy CDFs for these experiments with the maximum detection distance fixed at 3 cells to allow for sole observation of the effect of varying false negative rates. We first observe that false negative rates of 0.05 and 0.1 appear to shift UMD localization performance down by roughly 5% at each accuracy level. However, increasing the false negative rate from 0.1 to 0.2 in this region or in San Francisco yields virtually no decrease in localization accuracy, suggesting some method resiliency to further increases in this rate. The trends described here and in following analyses hold for classifiers with lower detection distances, but are omitted for brevity.

5.3 The Effects of False Positives

We next modeled the effect of false positives on victim localization. We hypothesized that false positives would be more damaging to

the localization accuracy than false negatives. This is because in each half hour simulation, less than 5 unique buses are observed on average in either region, so there is far greater opportunity for false positives to occur compared to false negatives over the simulation lifetime. We thus modeled classifiers with lower false positive rates of 0.01, 0.05, and 0.1 yet no false negatives for each maximum detection distance for each region. Figure 2(c) shows the localization accuracy CDFs for these experiments with the maximum detection distance fixed at 3 cells as before.

For UMD, we see an immediate performance impact when the false positive rate is set at just 0.01. Both rates of 0.01 and 0.05 seem to have similar effects on performance, lowering the proportion of simulations which achieve exact localization by 5-8% and lowering the proportion of simulations predicting a location within 80m of the victim by roughly 10%. A higher rate of 0.1 greatly decreases within 80m accuracy compared to a perfect classifier, dropping the cumulative proportion of simulations achieving that accuracy by 20%. These patterns hold for San Francisco as well, save for the experiment with a false positive rate of 0.01 which performs no differently than a perfect classifier.

Comparing these CDFs to those for false negatives, it appears false positive rates of 0.05 or 0.1 overall have only a slightly worse effect on localization accuracy than false negative rates of the same values. We theorize however that increasing false positive rates to 0.2 as we tested for false negatives would continue to degrade localization performance based on our observations, whereas increasing false negative rates from 0.1 to 0.2 had little effect in either region.

5.4 Classifiers with False Negatives and Positives

We finally model classifiers with both non-zero false positive and false negative rates. Figure 2(d) displays the localization accuracy CDFs for the 3 increasingly imperfect classifiers we test in both regions compared to the perfect classifier baseline, again fixing the maximum detection distance at 3 cells. For UMD, we see that increasing the false positive rate to 0.01 and false negative rate to 0.05 immediately causes significant accuracy loss. A classifier with these low error rates locates 10% less victims to their exact location and 12% less within 80m than a perfect classifier. Increasing error rates further continues to decrease overall performance.

For San Francisco, we strangely see that a classifier with a 0.01 false positive rate and a 0.05 false negative rate appears to perform similarly to the worst one we model with rates of 0.1 and 0.2 respectively. We believe this is due to a surprisingly large percentage of simulations for the (0.01, 0.05) classifier in which a bus never passes the victim, which shifts the CDF for this experiment downwards. Only 59% of simulated victims were passed by a bus in that experiment, whereas close to 70% of simulated victims were within range of a bus for each of the other experiments. We thus view this experiment as an outlier for how a classifier with error rates of 0.01 and 0.05 and a max detection distance of 3 cells would perform and believe running the experiment again would produce a CDF shifted upwards by roughly 10%.

This experiment aside, we see from the other San Francisco CDFs that classifiers with error rates of (0.05, 0.1) or (0.1, 0.2) negatively impact localization performance as they do in UMD compared to a perfect classifier. In fact, the worst imperfect classifier only achieves roughly half the localization accuracy of a perfect classifier in either region. It is clear then from these experiments that an attacker implementing a covert bus detection mechanism within a malware app should strive to reduce both false positive and false negative rates as much as possible to avoid significant localization accuracy loss.

5.5 Faster Localization

We next examine how quickly victims can be localized with this method. Recall that our experiment simulations simply returned the most probable location of the victim after half an hour of observations. We do not evaluate how waiting for further observations might affect localization accuracy, but we believe that longer observation periods will only increase the number of victims who are passed by a bus and lead to increased localization accuracies. Instead, we explore whether some victims can be located faster than after 30 minutes using an early stopping heuristic. It's important to note that an aggressive early stopping heuristic may hurt the method's localization accuracy, returning victim location predictions which may have been improved upon by waiting for further observations. We thus choose a conservative heuristic; if a cell in the region grid reaches a probability over 0.5, (meaning that cell has a greater probability than all other cells in the grid combined) that cell is immediately returned as the victim's predicted location.

By saving the cell with the highest probability in the grid along with that probability in each simulation, we can retroactively apply our early stopping heuristic to all simulations to observe its effects on localization time and accuracy. We find that of the 3207 simulations with locatable victims in UMD experiments, 1102 (34.7%) could stop early via this heuristic. The average stopping time for these simulations was 16 minutes and 10 seconds, a nearly 50% time decrease. Only 57 early stopped simulations (5.1% of them) return a prediction which is worse than the prediction otherwise returned at the end of the half hour. We find similar results for San Francisco. For the 3228 simulations with locatable victims in those experiments, 966 (30.1%) could stop early. The average stopping time for these simulations was 15 minutes and 27 seconds. Here 89 early stopped simulations (8.9% of them) return a worse prediction. For both regions then, we find that over 30% of simulations in which a bus passes the victim can be stopped after 15 or 16 minutes using our heuristic, causing a loss in localization accuracy in less than 10% of them. We therefore believe this heuristic could be used in practice, but leave open the possibility for better early stopping criteria to be developed.

5.6 Defenses

This localization method inherently relies upon the ability to leverage publicly available APIs to retrieve the real-time locations of public transit vehicles. Removing public access to such an API would thus defeat the attack, but it would also prevent transit system users from learning where buses are. Requests to an API might instead be rate limited to prevent constant bus location downloads by an attacker, but an attacker with multiple IP addresses could balance requests around these limits to evade this defense. We nonetheless believe that vehicle location API providers can limit data available

to an attacker to defeat this attack. We first recommend that bus locations not be updated at a high frequency. If bus locations are only updated every 30-50 seconds, it would be difficult for an attacker to pinpoint where any bus truly is at a given time to accurately update the region grid. Additionally, we recommend that other information which could be used to guess a bus's exact location based on other information be removed or have noise added. For example these APIs often report the time it will take a bus to reach the next stop on a route and the bus's speed. Ensuring that bus arrival estimates have low granularity (in minutes instead of seconds) and not reporting each bus's current exact speed and heading would make it more difficult for attackers to extrapolate bus positions from 30+ second old location information.

User diligence in preventing malicious applications from accessing unneeded permissions may also limit this and similar side-channel attacks. Users often do not understand the significance or functionality of permissions that apps request (Felt et al), so greater user education may prevent the proliferation of malicious apps which take advantage of excessive permissions.

6 CONCLUSION AND FUTURE WORK

Our experiments demonstrate that the ability to covertly detect when buses are nearby a stationary smartphone user is often enough information to locate the user within a city or university with a bus tracking API. Our localization method was able to locate the majority of simulated victims who were passed by at least one bus to within 80m of their actual location in all experiments, even those with relatively high false negative and false positives rates. Moreover, on average 26% of UMD victims and 22% of San Francisco victims could be located to their exact 20m by 20m area across all experiments. We find that modest false negative and false positive error rates in bus detection will degrade localization accuracy, but they do not prevent localization altogether. Additionally, victims can sometimes be located as quickly as in 15 to 16 minutes using a coservative early stopping heuristic.

In the future, we aim to explore whether non-stationary victims can be localized with similar degrees of accuracy. We also believe our experiment results warrant the creation and testing of real malicious bus detecting smartphone apps. There is also room for future work to define better early stoppage heuristics. Lastly, we are interested in exploring if leveraging a combination of public vehicle tracking APIs can be used to improve localization.

REFERENCES

[1] 2017. MTA Data Feeds. http://datamine.mta.info/. (April 2017).

[2] 2017. NextBus. http://webservices.nextbus.com/. (April 2017).

[3] 2017. NextBus Customers. http://nextbus.cubic.com/Our-Customers/Around-the-World. (April 2017).

[4] Ian Anderson and Henk Muller. 2006. Context Awareness via GSM Signal Strength Fluctuation. *Pervasive '06: Proceedings of the 4th International Conference on Pervasive Computing, Late breaking results* 01 (2006), 27–31. http://www.cs.bris.ac.uk/Publications/Papers/2000528.pdf

[5] Martin Azizyan, Romit Roy Choudhury, and Ionut Constandache. 2009. SurroundSense : Mobile Phone Localization via Ambience Fingerprinting. *MobiCom '09* (2009), 261–272. https://doi.org/10.1145/1614320.1614350

[6] R. Biswas and S. Thrun. 2004. A passive approach to sensor network localization. *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)* 2 (2004), 1544–1549. https://doi.org/10.1109/IROS.2004.1389615

[7] Wei Dai. 2017. Acoustic Scene Recognition with Deep Learning. (2017), 1–18.

[8] Boris Defreville, FranÃğois Pachet, Christophe Rosin, and Pierre Roy. 2006. Automatic Recognition of Urban Sound Sources. In *Audio Engineering Society Convention 120.* http://www.aes.org/e-lib/browse.cfm?elib=13631

[9] Lauren Frayer. 2015. Free Wi-Fi On Buses Offers A Link To Future Of 'Smart Cities'. (Mar 2015). http://www.npr.org/sections/alltechconsidered/2015/03/02/389250795/free-wi-fi-on-buses-offers-a-link-to-future-of-smart-cities

[10] Jun Han, Emmanuel Owusu, Thanh-le Nguyen, Adrian Perrig, and Joy Zhang. ACComplice : Location Inference using Accelerometers on Smartphones Loca4on Inference Background : Trajectory Inference Loca4on Inference. (????), 5.

[11] Fabian Hoflinger, Rui Zhang, Joachim Hoppe, Amir Bannoura, Leonhard M. Reindl, Johannes Wendeberg, Manuel Buhrer, and Christian Schindelhauer. 2012. Acoustic Self-calibrating System for Indoor Smartphone Tracking (ASSIST). *2012 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2012 - Conference Proceedings* November (2012), 13–15. https://doi.org/10.1109/IPIN.2012.6418877

[12] Thomas Janson, Christian Schindelhauer, and Johannes Wendeberg. 2010. Self-localization application for iPhone using only ambient sound signals. *2010 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2010 - Conference Proceedings* September (2010), 15–17. https://doi.org/10.1109/IPIN.2010.5648094

[13] J. Krumm and E. Horvitz. 2004. LOCADIO: inferring motion and location from Wi-Fi signal strengths. In *The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004.* 4–13. https://doi.org/10.1109/MOBIQ.2004.1331705

[14] Yan Michalevsky, Dan Boneh, Aaron Schulman, and Gabi Nakibly. 2015. PowerSpy: Location Tracking using Mobile Device Power Analysis. *Usenix Security* (2015). arXiv:1502.03182v2

[15] Kavitha Muthukrishnan, Berend Jan Van Der Zwaag, and Paul Havinga. 2009. Inferring motion and location using WLAN RSSI. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 5801 LNCS (2009), 163–182. https://doi.org/10.1007/978-3-642-04385-7_12

[16] Roman Schlegel, Kehuan Zhang, and Xiaoyong Zhou. 2011. Soundcomber: A stealthy and context-aware sound trojan for smartphones. *Proceedings of the 18th Annual Network and Distributed System Security Symposium (NDSS)* (2011), 17–33. https://www.cs.indiana.edu/

[17] Timothy Sohn, Alex Varshavsky, Anthony LaMarca, Mike Y. Chen, Tanzeem Choudhury, Ian Smith, Sunny Consolvo, Jeffrey Hightower, William G. Griswold, and Eyal de Lara. 2006. *Mobility Detection Using Everyday GSM Traces.* Springer Berlin Heidelberg, Berlin, Heidelberg, 212–224. https://doi.org/10.1007/11853565_13

[18] Sarah Whitten. 2016. New NYC buses to feature free Wi-Fi, USB ports. (Mar 2016). http://www.cnbc.com/2016/03/10/new-nyc-buses-to-feature-free-wi-fi-usb-ports.html

[19] Xiaoyong Zhou, Soteris Demetriou, Dongjing He, Muhammad Naveed, Xiaorui Pan, XiaoFeng Wang, Carl a. Gunter, and Klara Nahrstedt. 2013. Identity, location, disease and more: inferring your secrets from android public resources. *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security - CCS '13* (2013), 1017–1028. https://doi.org/10.1145/2508859.2516661

[20] Yajin Zhou. 2012. Dissecting Android Malware : Characterization and Evolution. 4 (2012). https://doi.org/10.1109/SP.2012.16

APPENDIX A FULL EXPERIMENT RESULTS

The tables on the following pages detail the localization accuracies achieved in each experiment we conducted. Each row corresponds to an experiment, with the first three columns listing the classifier parameters chosen for the experiment. The Proportion Passed by Bus column denotes the percentage of simulated victims who could have been localized in each experiment given that a bus passed them at some point during the simulations. The final columns denote the percentage of victims who were located to within a certain distance from their actual location, similar to the CDFs in Figure 2.

Max Detect Dist.	False Positive Rate	False Negative Rate	Prop. Passed by Bus	Prop. Exactly Correct	Prop. Within 1 Cell	Prop. Within 2 Cells	Prop. Within 3 Cells
1	0.00	0.00	0.46	0.22	0.31	0.41	0.41
1	0.01	0.05	0.46	0.31	0.37	0.42	0.42
1	0.01	0.00	0.49	0.26	0.32	0.44	0.44
1	0.05	0.10	0.46	0.20	0.27	0.35	0.35
1	0.05	0.00	0.48	0.28	0.35	0.41	0.41
1	0.10	0.20	0.47	0.20	0.25	0.29	0.29
1	0.10	0.00	0.46	0.23	0.29	0.35	0.35
1	0.00	0.05	0.53	0.27	0.39	0.48	0.48
1	0.00	0.10	0.49	0.26	0.37	0.46	0.46
1	0.00	0.20	0.44	0.28	0.37	0.42	0.42
2	0.00	0.00	0.49	0.26	0.38	0.47	0.48
2	0.01	0.05	0.47	0.27	0.35	0.43	0.43
2	0.01	0.00	0.53	0.31	0.38	0.50	0.51
2	0.05	0.10	0.47	0.24	0.31	0.38	0.40
2	0.05	0.00	0.48	0.28	0.34	0.46	0.46
2	0.10	0.20	0.47	0.16	0.20	0.28	0.29
2	0.10	0.00	0.46	0.23	0.31	0.40	0.40
2	0.00	0.05	0.40	0.20	0.27	0.38	0.39
2	0.00	0.10	0.47	0.23	0.33	0.43	0.43
2	0.00	0.20	0.51	0.23	0.35	0.44	0.46
3	0.00	0.00	0.71	0.39	0.48	0.61	0.70
3	0.01	0.05	0.64	0.30	0.40	0.49	0.58
3	0.01	0.00	0.66	0.33	0.46	0.56	0.63
3	0.05	0.10	0.62	0.27	0.36	0.43	0.53
3	0.05	0.00	0.69	0.36	0.47	0.53	0.62
3	0.10	0.20	0.66	0.22	0.30	0.37	0.45
3	0.10	0.00	0.67	0.31	0.38	0.49	0.55
3	0.00	0.05	0.69	0.34	0.43	0.55	0.65
3	0.00	0.10	0.63	0.26	0.40	0.49	0.55
3	0.00	0.20	0.65	0.26	0.40	0.49	0.57

Table 3: UMD Full Experiment Results

Max Detect Dist.	False Positive Rate	False Negative Rate	Prop. Passed by Bus	Prop. Exactly Correct	Prop. Within 1 Cell	Prop. Within 2 Cells	Prop. Within 3 Cells
1	0.00	0.00	0.49	0.26	0.31	0.34	0.34
1	0.01	0.05	0.23	0.15	0.19	0.19	0.19
1	0.01	0.00	0.54	0.27	0.30	0.35	0.35
1	0.05	0.10	0.24	0.16	0.18	0.18	0.18
1	0.05	0.00	0.49	0.25	0.30	0.33	0.33
1	0.10	0.20	0.16	0.10	0.12	0.12	0.12
1	0.10	0.00	0.54	0.23	0.27	0.30	0.30
1	0.00	0.05	0.52	0.25	0.31	0.34	0.34
1	0.00	0.10	0.51	0.23	0.26	0.31	0.32
1	0.00	0.20	0.50	0.21	0.24	0.29	0.29
2	0.00	0.00	0.54	0.24	0.31	0.39	0.41
2	0.01	0.05	0.52	0.22	0.25	0.35	0.35
2	0.01	0.00	0.49	0.18	0.26	0.34	0.36
2	0.05	0.10	0.55	0.28	0.34	0.38	0.38
2	0.05	0.00	0.50	0.21	0.27	0.34	0.34
2	0.10	0.20	0.49	0.16	0.21	0.30	0.30
2	0.10	0.00	0.54	0.20	0.25	0.36	0.36
2	0.00	0.05	0.51	0.23	0.32	0.39	0.41
2	0.00	0.10	0.56	0.27	0.32	0.41	0.43
2	0.00	0.20	0.56	0.25	0.33	0.40	0.41
3	0.00	0.00	0.70	0.23	0.36	0.46	0.54
3	0.01	0.05	0.59	0.17	0.24	0.33	0.40
3	0.01	0.00	0.70	0.23	0.36	0.46	0.54
3	0.05	0.10	0.69	0.31	0.39	0.44	0.50
3	0.05	0.00	0.73	0.27	0.32	0.40	0.48
3	0.10	0.20	0.68	0.21	0.26	0.31	0.37
3	0.10	0.00	0.69	0.18	0.25	0.35	0.41
3	0.00	0.05	0.69	0.23	0.34	0.45	0.52
3	0.00	0.10	0.64	0.24	0.31	0.41	0.47
3	0.00	0.20	0.64	0.21	0.35	0.41	0.46

Table 4: San Francisco Full Experiment Results