THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

COLLEGE OF INFORMATION SCIENCES AND TECHNOLOGY

# A Framework to Optimize Geofence Privacy and Security

EILEEN PI
SPRING 2022

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Cybersecurity Analytics and Operations
with honors in Cybersecurity Analytics and Operations

Reviewed and approved* by the following:

Anna Squicciarini
Professor of Information Sciences and Technology
Thesis Supervisor

Marc Friedenberg
Assistant Teaching Professor of Information Sciences and Technology
Honors Adviser

* Electronic approvals are on file.

# ABSTRACT

Cellphones and smart devices are part of our lives and integrate seamlessly into our daily activities. Geolocation enabled applications are widely adopted and continue to increase in use and demand. Geofencing is a specialized geospatial intelligence tool that enables useful features in many applications, such as Google Maps, Uber, Covid-19 contact tracing, and Pokémon Go. The persistence of location reporting may place the privacy of users at an increased risk to reveal private personal information and movement trajectories based on location, time, and historical movement patterns. This research proposes a geofencing cybersecurity framework which offers a dynamic, quantitative, and scalable obfuscation algorithm supporting customizable levels of privacy based on person, time, and place. This geofencing cybersecurity framework overcomes the tradeoff between individual privacy and organizational security and offers the ability to harmonize and maximize both privacy and security. Using this geofencing obfuscation algorithm, individuals can increase their privacy while community and company can increase the security of their assets, mission, and values.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# ACKNOWLEDGEMENTS

I want to acknowledge and appreciate all the support, assistance, and guidance I received throughout the writing of this thesis.

I would like to thank my thesis supervisor, Anna Squicciarini, for all her encouragement and abundant knowledge which sparked my interest in privacy and geolocation. I want to thank Professor Squicciarini for her patience and guidance.

I would also like to thank Primal Pappachan, a Postdoctoral Scholar working with Professor Squicciarini, for his insightful comments, feedback, and suggestions that pushed me to sharpen my thinking and brought my work to a higher level.

I am deeply grateful to my honors advisor, Marc Friedenberg, who is a great advisor and professor. Thank you for encouraging me to take on this challenge and persist in excellence.

I would like to express my sincere gratitude to my family for their continual love, patience, unwavering support, and being a constant voice of encouragement throughout my four years of college. Without their tremendous understanding and encouragement in the past year, none of this would have been possible. I would like to extend my thanks to my dad for walking along-side, praying for, and uplifting me.

I also want to thank everyone who prayed for me, especially Sandy Mackin, Rachel Yeung, and Stephanie Xu, for walking with me through difficult seasons and encouraging me to continue my faith journey.

Thank you God for being my strength when I felt weak.

# Chapter 1

## Introduction

Location based services and geofencing have been increasingly utilized on a day-to-day basis and is a key service to many applications and services to facilitate traveling, marketing, entertainment, security, geo-intelligence, and human resource management [9]. However, due to the advancement and integration of these technologies in our daily lives, the derived geo-spatial intelligence is an inherent risk to compromise the privacy of users. Location data can be used to determine a personally identifiable information (PII), private identity, and sensitive personal information and activity [11]. Even with efforts to anonymize user identities and subscriptions, PII can still be revealed based on integrated and shared information and may cause violation to individual's digital and physical privacy. In capable hands, these technologies can ingest large amounts of raw data information and collated and analyzed into actionable intelligence. This may be advantageous and useful for legitimate organizations or become dangerous and malicious tools for attackers ([9], [12]).

### Research Objective

This thesis intends to implement a design using geofencing to demonstrate the integration of multiple informational categories and levels of privacy. This framework takes into account the various levels of privacy necessary to individuals, families, organizations, and the government. By enabling both individuals and organizations to customize privacy levels based on the

categories of person, time, and place, this will maximize the individual privacy and organizational security at the same time.

# Chapter 2

# Literature Review

## Location Based Services

Location based services (LBS) have been increasingly utilized on a day-to-day basis and is a key service supporting many smart-device and computer applications. For example, companies have found ways to apply geospatial information as resource management, travel information, restaurant recommendations, rideshare, advertising, weather forecasts, mobile games, and more ([4], [5]). LBS can also be used for security and safety purposes, such as when crimes and natural disasters occur, sharing your location to people you trust during crisis situations, roadside assistance, loss recovery, and fraud prevention. When inappropriately utilized, LBS can be misused or even facilitate unethical or criminal activities such as disclosure of private, sensitive, or classified information ([9], [12], [6]).

The raw data from LBS have commercial value to advertisers, companies, and malicious entities like commercial competitors or national-level cyber actors. At personal level, LBS may be utilized to satisfying intrusive curiosities or invasion of privacy. For businesses, LBS may be patterned to disclosed proprietary information and commercial activities [12]. For a user of a geolocation application, such as Google Maps, Uber, Covid-19 contact tracing, and Pokémon Go, the technology is set to persistently identify and collect the user's real-time geolocational data. LBS can easily collect, organize, and analyze these lakes of big data to perform variety of real-time and retrospective services and functions [11]. Furthermore, LBS can proactively and accurately predict critical future trends based on patterning of human locational activities and behavior leading to early risk mitigation [1].

### *GPS, Wi-Fi, Cellular Data, Bluetooth, and RFID*

LBS uses real-time data from a smartphone's global positional system (GPS), Wireless Fidelity (Wi-Fi), Cellular signals, Bluetooth technology, and radio frequency identification (RFID) to accurately track and identify a location of a user or asset. The LBS resolution varies depending on the specific technology such as GPS satellites, Wi-Fi networks, and cell towers [5]. An integrated approach using all or combination of these technologies enhances the LBS resolution and coverage areas and structures. Typically, the users of the devices have the option to opt-in to use the location services on their devices. The location services will identify, collect, and report the locational data and enable other geospatial services such as geolocation, geosensing, geotargeting, geofencing, and geointelligence. [14]. This research focuses on geofencing application.

## Geofencing

### *Definition*

Geofencing is a location-based service that triggers a pre-programmed action when a device enters, stays, and exits a designated physical boundary/perimeter, known as a geofence. Depending on the geofence configuration, the application can trigger a response such as pushing notifications, targeted advertisements, security alerts, activity logging, and more. Although commonly used for mobile apps, geofencing can be used for control and tracking people, equipment, inventories, vehicles, and drones. Geofences can be set to a specific place, structure, demographic market area, business category, brand location, and political jurisdictions ([10], [7]). It can also be used to monitor public areas as well as private secure areas.

### *Technological Requirements*

For geofencing to work, a comprehensive map of physical structures and point of interests must be first defined. This is followed by a survey of the areas and structures to define the specialized meaning or value for the geofenced area and its boundaries. Lastly, a technology-enabled integrated platform is needed to detect real-time geolocation data (such as a GPS sensor to detect geolocation data), transmit geolocation data (such as a cellular phone or smart-device), collect the data (such as Google, PSU, or military servers), process data based on pre-defined geofencing algorithms (server contains geofencing services and associated algorithms), and finally execute pre-defined sets of actions (such as assignment of priority or privacy level). Geofencing uses the raw data from geolocation software to detect movement activities and correlate to the boundary defined for a location of interest. The geofence will detect, analyze, and trigger a response when the device enters, stays, or exits the specified area.

### *Examples of Geofencing*

Here are two examples of real-life application of geofencing. First, geofencing has been commonly associated and used for marketing and retail purposes. Using location-based intelligence, advertisement can be crafted to direct users to nearby competing stores or be tailored to offer incentive for a particular store. By connecting with users, stores can provide targeted information to individuals based on their real-time geographic location through mobile apps or webpages ([15], [7]). This is easily accomplished; stores create a geofence in a designated area and an ad campaign for the geofence. When a user steps into the specified geofence, the ads or promotions are pushed and delivered to the user via notification, in-app ads, banner, search, or display ads [15]. This can be extremely useful when it comes to marketing, social media, and boosting profit in businesses, as well as gaining advantage over commercial competition. Businesses can engage and deliver personalized promotions and targeted

advertisements to consumers that enter a geographical range of the store or even a competitor's store ([7], [8]).

The second example of real-life application of geofencing is security tracking. Geofencing can be configured to trigger alerts and used to monitor, control, and collect activity information in secure locations [13]. User locations can be continuously reported. It can be used to automate timecards as in human resource management. It can also be utilized to track company equipment and vehicles when they enter and exit the specific geofence areas for risk mitigation and management [13]. This can be used for organizational security and asset management to improve equipment tracking, device logging, and theft prevention. For example, RFID-enabled devices provide real-time alerts when a tagged asset or equipment enters or exits a designated gate or perimeter.

**Location Obfuscation**

Location obfuscation is a technique used in location-based services to increase the level of locational privacy of an individual by altering, substituting, or generalizing their location such that it discloses the location of the individual at the desired level of resolution in detail and accuracy ([11], [12]). The challenge in location obfuscation lies with the balance between privacy protection and technical accuracy that is needed to contain enough details to provide a satisfactory service. Some common techniques include spatial cloaking, randomization, invisible cloaking, adding noise, rounding, and redefining the possible areas of location ([3], [11], [12]). The purpose of location obfuscation is to distort and generalize the individual's locational information to prevent malicious tracking of an individual's trajectory [11].

**Location Privacy Concerns**

Although LBS has been highly transformative and widely accepted into everyone's daily lives, there has been concerns about an individual's privacy and safety. LBS can collect, organize, and analyze lakes of big data to perform variety of real-time and retrospective analysis which can determine and predict past, current, and future human locational activities and behaviors [11]. Geofence can determine patterns of movements and temporal relationships (entry, exit, and stay durations) relative to a point of interest. The exposure of location data can reveal highly sensitive individual attributes such as home addresses, work locations, religious affiliation, health, political orientation, and social relationships ([1], [11], [6]). For example, The work location can be determined based on regular weekday stay at a location. The home location can be determined based on the daily and nightly stay at a location. Even the private health conditions, such as having cancer or pregnancy, of a user can be easily inferred based on the frequency and duration of visits to a cancer or pregnancy related medical facilities. Another example includes the inference of an occupation, such as a foreign affair service ambassador or diplomat, from the frequency of visits to the embassy, government bases, and State Department. This can be concerning as certain occupations are prime targets for protests, bombings, intelligence, and political turmoil. When utilized appropriately, this technology is transformative in improving life experience, but it can also open new vulnerabilities and violations of the user's privacy and advance malicious purposes and causes ([12], [6]).

**Levels of Privacy**

Individuals and organizations have different requirements regarding location privacy and security ([6], [11]). For example, homes, shopping malls, corporate offices, government buildings, and military bases all have different requirements. Most existing privacy protection mechanisms use a static uniform privacy parameter, which may result in both inadequate security and insufficient privacy protection [6]. A

single static uniform privacy parameter does not take into consideration the specific requirements of a particular person, time, and place and may result in mis-regulated collection or protection of the type of data [2]. This can result in inefficient and excessive data collection. At best, there is waste of data storage. At worst, data can be sold or stolen with no limitation on the malicious ways it could be utilized.

Even within the same organization, the requirements may vary depending on the person, time, and places [6]. Some areas may require higher security while other areas can benefit from more public access. This needs to be balanced and maximized with the levels of expectation of privacy [11]. Users should be able to customize a privacy profile based on person, time, and place.

# Chapter 3

# Methodology

## Objectives

This study aims to protect smart device users' location privacy and promote organizational security through a geofencing obfuscation algorithm. Attackers can target a user and may even be able to de-anonymize the user based on the trajectory and mobility pattern. In order to properly obfuscate the user location, a well-defined level of privacy is needed to determine the level of precision and generalization of the location data. This research proposes a geofencing framework which offers a dynamic, quantitative, and scalable obfuscation algorithm supporting customizable levels of privacy based on person, time, and place. This framework overcomes the tradeoff between individual privacy and organizational security and offers the ability to harmonize and maximize both privacy and security. Using this geofencing obfuscation algorithm, individuals can increase their privacy while community and company can increase the security of their assets, mission, and values.

## Assumptions, Dependencies, and Constraints

- This research has the following assumptions, dependencies, and constraints

## Assumptions

- Based on classic concept, there is a tradeoff between security and privacy
- The levels, accuracy, and precision of geolocation data will correlate inversely relation with the level of privacy of individuals

- Current static uniform approach on privacy level is inadequate to address the dynamic balance required between security and privacy
- The privacy values are artificially set by virtual stakeholders and their presumed significance and values

**Dependencies**

- Geolocation raw data are dependent on technologies such as GPS, cellular signal, Wi-Fi, Bluetooth, and RFID.
- Geofence services need the raw geospatial data from geolocation services.
- This research demo utilizes Google Maps, Android Studio, and the Google Geofencing API

**Constraints**

- Only three informational categories, person, time, and place, are utilized for demonstration purposes, but the framework is scalable to expand to cover additional categories. Individual preference may also be added as an additional category.
- The obfuscation factor is set arbitrarily for the Penn State's campus activities. Different geofence systems may utilize different obfuscation factors based on their specific needs.
- For the purpose of this demo, the shape of the geofence is circular whereas in real situation, they would be the exact shape.
- For the purpose of this paper, the privacy is referring to individuals and not program or facility.

## Implementation/Development

The interface is built utilizing the Android Studio software, Google Maps Platform, and the Google Geofencing API. The Android Studio platform is chosen because it is a free and open-source platform with an easily accessible library and has a flexible Gradle-based build system, integration with Google Play services, and a fast and rich featured emulator. The Google Maps Platform is utilized because it is familiar to general users, easy to use, widely adopted, and interoperable with many Android mobile platforms [4]. This application uses Google Maps as a geolocation data source and interacts by using API keys through Maps SDK for Android to run on Android-compatible device or emulator. The Google Geofencing API further analyzes and enhances the raw geolocation data from Google Maps into actionable intelligence.

## Privacy Level Implementation

This research paper proposes a geofencing framework which offers a dynamic, quantitative, and scalable obfuscation algorithm supporting customizable levels of privacy based on person, time, and place.

### *Dynamic Considerations*

The current common implementation using a single static uniform privacy level does not adapt to the potential change in circumstances. For variation due to persons, the privacy level for the same location but different employee roles may differ between a doctor and a patient in a hospital. For variation due to time, privacy level for the same person may differ during working hours versus personal private hours. For variation due to place, the privacy level for the same person may differ between a shopping mall

versus a secure military facility. A dynamic framework is needed to address the various combinations of these variables.

### *Quantifiable Privacy Value*

In order to implement privacy levels and security protections efficiently and effectively, the geofencing obfuscation algorithm has to be quantitative and not just qualitative. This research proposes a multifactor calculation assigned independent privacy value to each category of person, time, and place. A value of 1 serves as the base value. An enhanced level of privacy will take a value that is greater than 1 based on a multiplication relationship. A decreased level of privacy will take a value that is between 0 and 1. The final privacy value is derived by multiplying all three factor values. As a baseline example, if the privacy value for a person is 1, time is 2, place is 2, the final privacy value is equal to 4. The equation is as follows:

$$Final\ Privacy\ Value = Person\ * Time\ * Location$$

**Table 1: Privacy Levels**

| | *0-1* | *1* | *1+* |
|---|---|---|---|
| **Person** | Lower Privacy | Standard | Higher Privacy |
| **Place** | Lower Privacy | Standard | Higher Privacy |
| **Time** | Lower Privacy | Standard | Higher Privacy |

For example, a person may have a standard baseline privacy configuration for person is 1, place is 1, time is 1. If the person is a police officer, the role needs to have a precise location tracking during working hours for rapid response and safety purposes. A proper privacy values for the police officer during work hours: person is 1, place is 1, time is 0.2, the final privacy value is equal to 0.2. For a police officer afterwork expect higher privacy. A proper privacy value for the police officer afterwork: person is

1, place is 1, time is 2. A parent may choose to pinpoint the precise location of a child and assign a low privacy value for the child [2]. If the privacy value for a child is 0.5, place is 1, time is 1, the final privacy value is equal to 0.5. A person who may choose to have lower privacy when going to a shopping mall to allow better shopping experience. If the privacy value for the shopper is 1, place is 0.3, time is 1, the final privacy value is equal to 0.3.

The final multifactor privacy score is calculated by multiplying all the privacy factor variables. Higher privacy value corresponds to lower geolocation precision (more geolocation generalization). Lower privacy value corresponds to higher geolocation precision (less geolocation generalization)

**Obfuscation Factor**

There are three common formats to specify geolocation data using latitude and longitude coordinates. They are as following:

**Table 2: Common Geolocation Data Formats**

| | |
|---|---|
| DDD° MM’ SS.S” | Degrees, Minutes and Seconds |
| DDD° MM.MMM’ | Degrees and Decimal Minutes |
| DDD.DDDDD° | Decimal Degrees |

For the purpose of this geofencing obfuscation algorithm, the decimal degree formatting system is utilized to calculate and obfuscate an individual’s position. The decimal degree format is easiest to use since it is a format with single unit. Other formats may also be utilized with minor modifications in the algorithms to accommodate the additional units of degree, minutes, and seconds.

**Obfuscation Factor**

The obfuscation factor is a universal value used to calculate the geolocation generalization. The obfuscation factor is in the unit of decimal degree. For the purpose of demonstration, this research selects a static decimal degree of 0.0003, which is equivalent to 109.37 feet, based on the conversation calculation of $decimal\ degees = \frac{feet}{364{,}567.2}$. The obfuscation factor can be easily standardized to other values as needed. A larger obfuscating factor will result in a larger generalized area for any specific privacy level. A smaller obfuscating factor will result in a smaller generalized area for any specific privacy level.

**Table 3: Obfuscation Factor (Decimal Degrees)**

| *Obfuscation Factor (Dec)* | *Obfuscation Factor (ft)* |
|---|---|
| 0.0003 Decimal Degree | 109.37 feet |

The next step to obfuscation the geolocation is to determine the size of the generalized area, and this is done by multiplying the final privacy value and the obfuscation factor. For example, if the final privacy value is 1 and the obfuscation factor is 0.0003 decimal degrees, then the radius of the generalized area is 109.37 feet. If the final privacy value is 3 and the obfuscation factor is 0.0003 decimal degrees, then the radius of the generalized area is 328.11 feet.

Finally, the obfuscated location can be anywhere within the generalized area, and this is calculated using a randomization function between 0 and the radius of the generalized area.

The equation to determine the new longitude/latitude coordinates below are as follows:

$$Current\ Latitude \pm Randomized(0, Final\ Privacy\ Value * Obfuscation\ Factor)$$
$$= New\ Latitude\ Coordinate$$
$$Current\ Longitude \pm Randomized(0, Final\ Privacy\ Value * Obfuscation\ Factor)$$
$$= New\ Longitude\ Coordinate$$

$$(Where\ Final\ Privacy\ Value = Person\ * Time\ * Location)$$

$$(Where\ Radius\ of\ the\ Generalized\ Area = \ Final\ Privacy\ Value * Obfuscation\ Factor)$$

**Scalability**

This geofencing obfuscating algorithm is scalable to accommodate additional categories as needed. Currently, there are three categories: people, place, and time. Additional categories such as security clearance, role-based positions, or emergency exemptions may be added as needed. Furthermore, a change in the baseline obfuscation factor will result in universal change for all calculations. Whereas a change in the categorical privacy values allows for customization at group level. The multifactorial multiplication calculation allows for meaningful and harmonized balance among all factors. Lastly, the randomized function generates specified but unpredictable geofencing obfuscation.

**Use Case / Demo**

This demo application shows a use case of an undergraduate student who is attending The Pennsylvania State University. For Hamilton Hall, a student residence, the geofenced area for a student during the daytime can have the following configurations: student = 1, time = 1, and place = 1. The same geofenced area for a student during the nighttime can have the following configurations: student = 1, time = 3, and place = 1. For the building containing the Penn State Breazeale Nuclear Reactor, this geofenced area may have the following configuration 24-hours every day to maximize security: student = 1, time = 1, and place = 0.2.

First, this demo application obtains the necessary user permissions to access the actual geolocation data to monitor for geofence related triggering events. The geofence is defined by the stakeholders of the geographical point of interest. The categorization of information and designated privacy levels are set by stakeholders of the geographical point of interest (such as public safety, privacy

advocates, community leaders, and regulatory bodies). Based on the geofencing obfuscation algorithm, the geolocation raw data will be obfuscated to a generalized area to maximize privacy and security.

For the purpose of this demo, the system administrator clicks on a location on the map to create a geofence area. The program takes the inputted latitude and longitude coordinates and adjusts the proximity of the location with the pre-defined radius, creating the circular perimeter around the location of interest. Android recommends a minimum geofence radius of 100-150 meters to help account for location accuracy of typical Wi-Fi networks. The geographic area, size, and location can be altered at any time. The API has a limit of 100 active geofences per app, per device user. The application will notify and log if the Geofence has failed to create or if there are too many geofences created. Once the geofence is created, it is added to a geofence request, and the program can start monitoring the geofences. The geofence does not expire unless the device is rebooted, service data is cleared, or the app data is cleared or uninstalled.

Once the geofence is created, the Google Android Geofence API analyzes and reports three events: (1) whether the individual enters the geofence, (2) dwells - exists within the geofence for a given minimal period and (3) exits - moved out of the geofence.

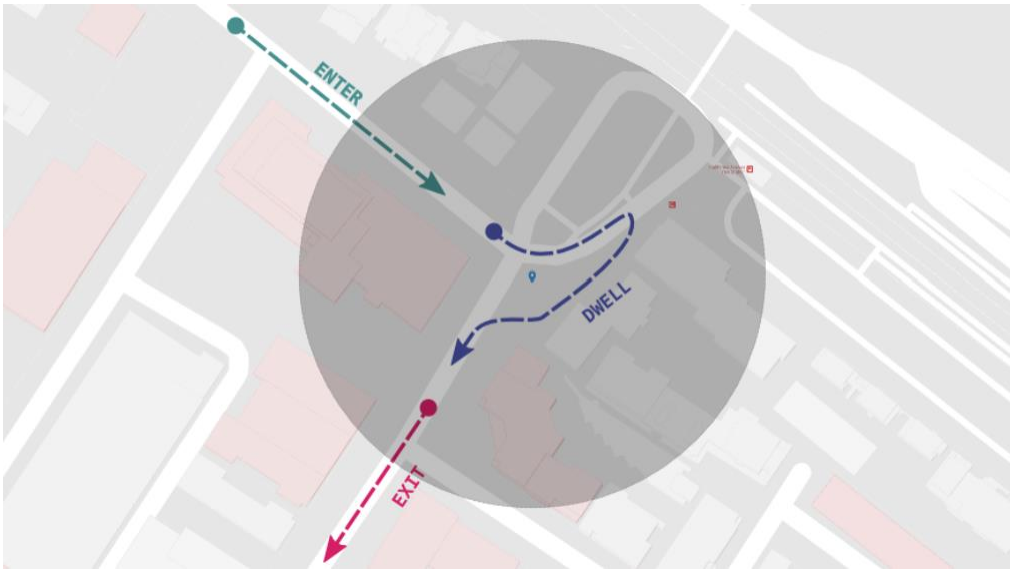


**Figure 1. Geofence Google API**

For evaluating the effectiveness of the proposed geolocation obfuscation algorithm, we have conducted multiple simulations where the individual, time, and place are located in The Pennsylvania State University.

***Demonstration #1 – Hamilton Hall Baseline***

The goal of this demonstration is to effectively model a geofenced area located in a student residence, Hamilton Hall, for a student during the daytime. The simulation relies on privacy framework and is conducted in multiple steps. The first step is to explicitly define the privacy policy parameters in the algorithm for the geofence with the following configurations: person = 1, time = 1, and place = 1. As shown in the left image in Figure 2, the system administrator should then click and create the geofence in the desired location, Hamilton Hall on the program application. In the middle image of Figure 2, Google Maps will prompt the user to allow permission to track their location.

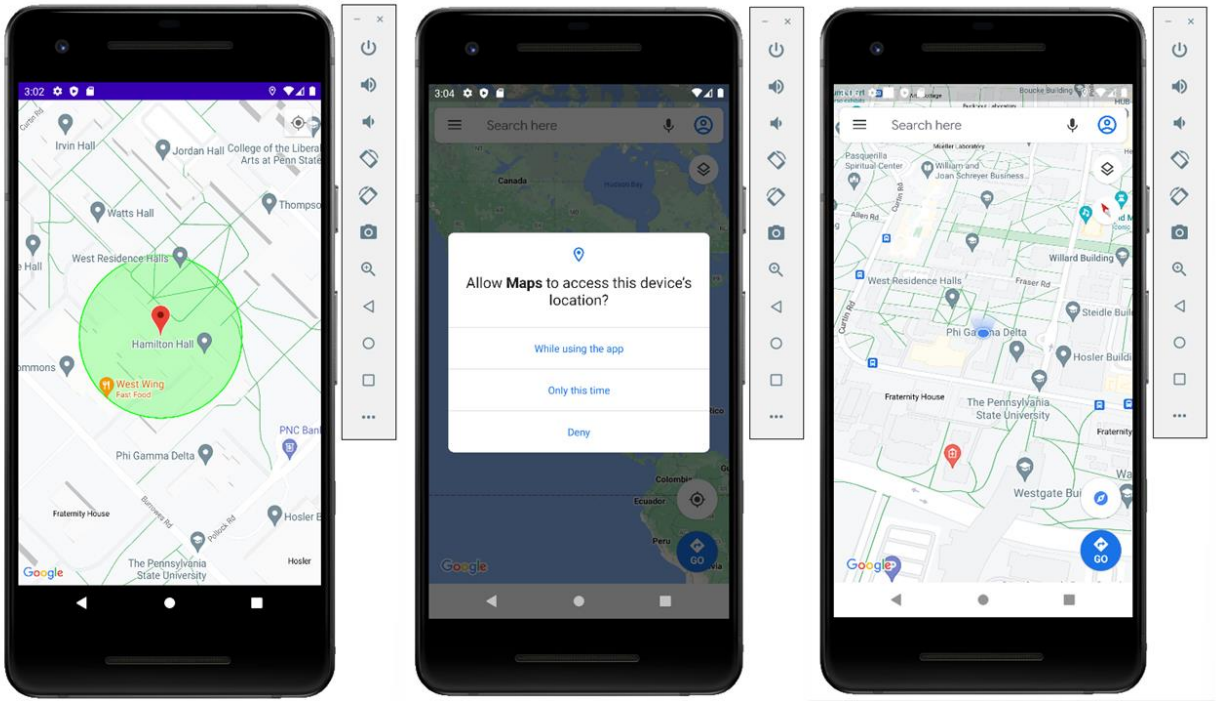


**Figure 2: Demo 1 – Hamilton Hall Baseline**

When the user enters the geofence, as displayed in the right image of Figure 2, the device will take the privacy function with the current latitude, longitude, final privacy value, and obfuscation factor to calculate the obfuscated location. Once this function execution is complete, the device will output a notification with the new location coordinates as demonstrated in Figure 3. These new coordinates factor the person, time, and place.

Old Coordinates: 40.7957655, -77.8666049

New Coordinates: 40.795249, -77.866487

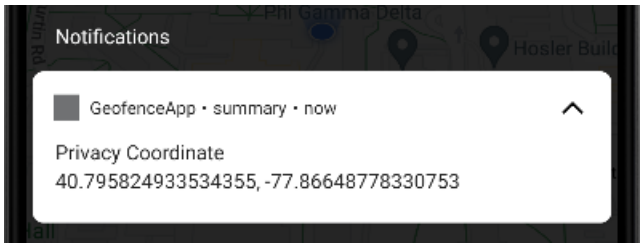


**Figure 3: Demo 1 – Obfuscated Location Notification**

***Demonstration #2 – Hamilton Hall Nighttime***

Another demonstration of the privacy policy framework can be used with the same student and location but different time. The geofenced area for a student in Hamilton Hall during the nighttime can have the following configurations: student = 1, time = 3, and place = 1, indicating higher privacy.

The simulation relies on privacy framework and is conducted in multiple steps. The first step is to explicitly define the privacy policy parameters in the algorithm for the geofence with the following configurations: person = 1, time = 3, and place = 1. As shown on the left image of Figure 4, the system administrator should then click and create the geofence in the desired location, Hamilton Hall on the program application. In the middle image of Figure 4, Google Maps will prompt the user to allow permission to track their location.

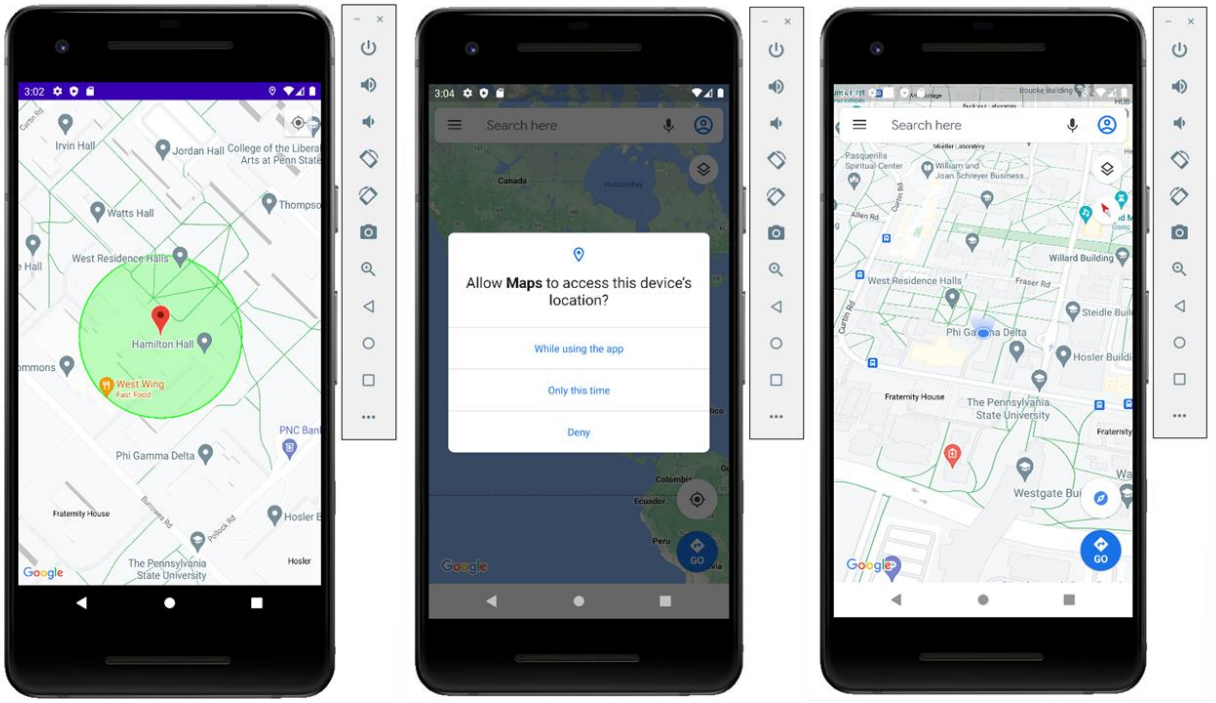


**Figure 4: Demo 2 – Hamilton Hall Nighttime**

When the user enters the geofence, as displayed in the right image of Figure 4, the device will take the privacy function with the current latitude, longitude, final privacy value, and obfuscation factor to calculate the obfuscated location. Once this function execution is complete, the device will output a notification with the new location coordinates as demonstrated in Figure 5. These new coordinates factor the person, time, and place.

Old Coordinates: 40.795767-77.8666046

New Coordinates: 40.796210, -77.865877

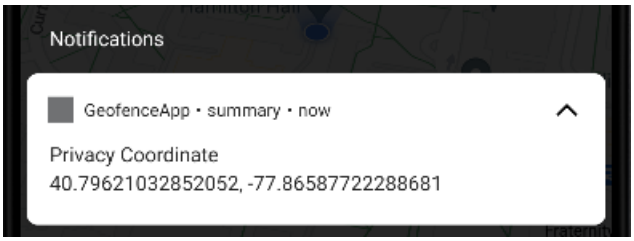


**Figure 5: Demo 2 – Obfuscated Location Notification (Higher Privacy, Larger Generalized Area)**

***Demonstration #3 – Breazeale Nuclear Reactor***

Another demonstration of the privacy policy framework can be used for the building containing the Penn State Breazeale Nuclear Reactor.

The simulation relies on privacy framework and is conducted in multiple steps. The first step is to explicitly define the privacy policy parameters in the algorithm for the geofence with the following configurations 24-hours every day to maximize security: student = 1, time = 1, and place = 0.2.

As shown in the left image of Figure 6, the system administrator should then click and create the geofence in the desired location on the program application. In the middle image of Figure 6, Google Maps will prompt the user to allow permission to track their location.

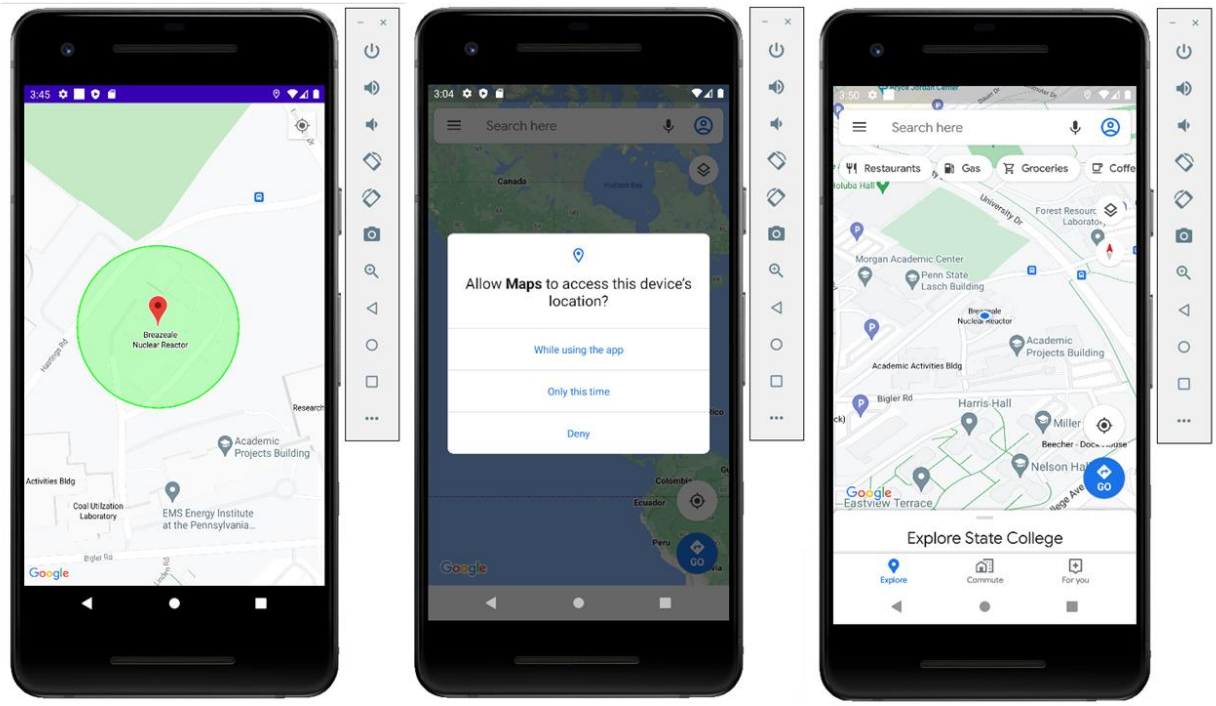


**Figure 6: Demo 3 – Breazeale Nuclear Reactor**

When the user enters the geofence, as displayed in the right image of Figure 6, the device will take the privacy function with the current latitude, longitude, final privacy value, and obfuscation factor to calculate the obfuscated location. Once this function execution is complete, the device will output a

notification with the new location coordinates as demonstrated in Figure 7. These new coordinates factor the person, time, and place.

Old Coordinates: 40.80390, -77.85350

New Coordinates: 40.803886, -77.85351

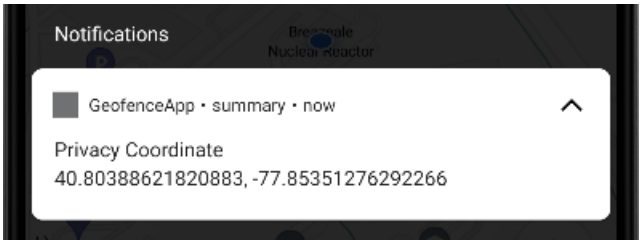


**Figure 7: Demo 3 – Obfuscated Location Notification (Less Privacy and Smaller Generalized Area)**

# Chapter 4

## Results

This research's geolocation obfuscation algorithm takes into consideration both the accuracy of location measurements and the users' needs of privacy. This research paper implements a geofencing cybersecurity based framework which offers a dynamic, quantitative, and scalable obfuscation algorithm supporting customizable levels of privacy based on person, time, and place and integrate and harmonize these categorical variables.

The interface is built utilizing the Android Studio software, Google Maps Platform, and the Google Geofencing API. This geofencing obfuscation algorithm is demonstrated by the use of three micro-services connected through API. Google Maps is the geolocation data source and interacts with Maps SDK for Android, and the raw data are analyzed and enhanced by Google Geofencing API into actionable intelligence.

The current static uniform privacy level does not adapt to the potential change in circumstances. This research implements a multifactorial calculation to dynamically determine privacy level by using independent privacy values assigned to each category of person, time, and place. Although other common geospatial formats can be utilized, this research uses the decimal degree formatting system to calculate and obfuscate an individual's position. This dynamic system is flexible and adaptable to various local categorical needs; furthermore, this research implements the obfuscation factors as a universal expectation for geospatial privacy. An increased value in obfuscation factor equals an increase in spatial privacy for any level of privacy. A change in obfuscation factor value will result in universal change for all calculations. The following equations are used to determine the generated the generalized obfuscated longitude and latitude coordinates.

$$Current\ Latitude \pm Randomized(0, Final\ Privacy\ Value * Obfuscation\ Factor)$$

$$= New\ Latitude\ Coordinate$$

$$Current\ Longitude \pm Randomized(0, Final\ Privacy\ Value * Obfuscation\ Factor)$$

$$= New\ Longitude\ Coordinate$$

$$(Where\ Final\ Privacy\ Value = Person\ * Time\ * Location)$$

$$(Where\ Radius\ of\ the\ Generalized\ Area = \ Final\ Privacy\ Value * Obfuscation\ Factor)$$

This multifactorial multiplication calculation allows for meaningful and harmonized balance among all factors. This research demonstration, which only selects and utilizes three categories, exemplifies the scalability of this dynamic and quantifiable geofencing obfuscation algorithm. It should be noted that this geofencing cybersecurity based framework for privacy is easily scalable to expand to cover additional categories; especially, individual preference may also be added as an additional category. When individual preference factor is combined with properly selected obfuscation factor, the individual and community expectation of privacy can be balanced and harmonized.

# Chapter 5

## Conclusion

The rapid growth in the reliance on geolocation services poses risks of an individual's geospatial privacy. This paper implements an efficient and effective geofencing obfuscation algorithm that protects the geospatial privacy of individuals and maximizes the security of organization and community. This framework takes into account three privacy categories based on person, time, and place and shows that the approach is dynamic, quantifiable, and scalable. Geofencing is already widely adopted and used in our daily activities, such as Google Maps, Uber, Covid-19 contact tracing, and Pokémon Go. By implementing this privacy preserving framework, it will enable the confidentiality, integrity, and availability of individual privacy and organizational security. This framework can improve the users' life experience, accurate patterning of human behavior, early detection of critical trends, risk mitigation proactively, reactively, and real-time [1].

## Limitations

When designing a location-privacy preserving mechanism, there are various tradeoffs, such as security, confidentiality, integrity, and availability. In order to achieve a higher level of location privacy, the mechanism may sacrifice accuracy and quality [6]. The greater the obfuscation applied to the location, the wider the generalized area can deviate from the user's real position. [6].

There are two main limitations when extreme small or large privacy values are used. An extreme small or large privacy value in a category may impact and offset the overall balance of the total privacy value. An extreme small privacy value will not improve precision and the final accuracy of the geospatial data. Ultimate precision is based on the device limitation. An extreme large privacy value can be useful and important for an individual's privacy and security but can trivialize and negate all other categorical values.

Another limitation occurs if there is a compromise at the device level. When the geolocation device itself is compromised, the adversary can obtain unauthorized access the raw geospatial data directly. This type of compromise circumvents the geofence obfuscation algorithm.

The privacy defined in this research refers to people privacy and not program or building privacy.

## Improvements and Future Research

### Geofence Boundaries

There most common methods of defining the geofence boundaries are using radii and polygonal geofences. This application utilized a circular and radius geofence. There are blind spot areas that are captured which are not intended versus defining the exact shape of the area. For example, most buildings are not circular so the geofence may capture areas around the square building that you do not intend to. The actual implementation should implement polygonal geofences which possess better accuracy than radius based geofences.

### Person-Specific Privacy

As discussed in this research paper, this study can be extended to allow individuals to define their own privacy levels. This allows each user to specify their own personal preference. Other personal categorical variables can also be developed based on the person-specific patterns. The user could collect their own patterns and trajectories using automated dataset collectors to determine patterns over a period of time.

# Appendix A

## java/MainActivity

```
package com.example.geofenceapp;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

## java/MapsActivity

```
//Adapted from https://github.com/trulymittal/Geofencing
package com.example.geofenceapp;

import androidx.annotation.NonNull;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import androidx.fragment.app.FragmentActivity;
import android.Manifest;
import android.annotation.SuppressLint;
import android.app.PendingIntent;
import android.content.pm.PackageManager;
import android.graphics.Color;
import android.os.Build;
import android.os.Bundle;
import android.util.Log;
import android.widget.Toast;
import com.google.android.gms.location.*;
import com.google.android.gms.maps.*;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.CircleOptions;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.example.geofenceapp.databinding.ActivityMapsBinding;
```

```
public class MapsActivity extends FragmentActivity implements
OnMapReadyCallback, GoogleMap.OnMapLongClickListener {

    private static final String TAG = "MapsActivity";
    private GoogleMap mMap;
    private GeofencingClient geofencingClient;
    private GeofenceHelper geofenceHelper;

    private int FINE_LOCATION_ACCESS_REQUEST_CODE = 10001;
    private int BACKGROUND_LOCATION_ACCESS_REQUEST_CODE = 10002;

    private float GEOFENCE_RADIUS = 50;
    private String GEOFENCE_ID = "SOME_GEOFENCE_ID";

    private ActivityMapsBinding binding;
    private LatLng hold;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        binding = ActivityMapsBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        // Obtain the SupportMapFragment and get notified when the map is
ready to be used.
        SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);

        geofencingClient = LocationServices.getGeofencingClient(this);
        geofenceHelper = new GeofenceHelper(this);
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;

        LatLng stateCollege = new LatLng(40.7982, -77.8613);
        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(stateCollege, 16));
        enableUserLocation();
        mMap.setOnMapLongClickListener(this);

        if(hold == null)
            hold = new LatLng(40.7938366, -77.8676468);
        addCircle(hold, this.GEOFENCE_RADIUS);
    }

    private void enableUserLocation(){
        if(ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) ==
                PackageManager.PERMISSION_GRANTED) {
            mMap.setMyLocationEnabled(true);
        }
        else{
```

```
            //Ask for permission
            if(ActivityCompat.shouldShowRequestPermissionRationale(this,
                    Manifest.permission.ACCESS_FINE_LOCATION)){
                ActivityCompat.requestPermissions(this, new String[]
{Manifest.permission.ACCESS_FINE_LOCATION},
FINE_LOCATION_ACCESS_REQUEST_CODE);
            }
            else{
                ActivityCompat.requestPermissions(this, new String[]
{Manifest.permission.ACCESS_FINE_LOCATION},
FINE_LOCATION_ACCESS_REQUEST_CODE);
            }
        }
    }

    @SuppressLint("MissingPermission")
    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions, @NonNull int[] grantResults) {
        if (requestCode == FINE_LOCATION_ACCESS_REQUEST_CODE) {
            if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                mMap.setMyLocationEnabled(true);
            }
            else {
            }
        }
        if (requestCode == BACKGROUND_LOCATION_ACCESS_REQUEST_CODE) {
            if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                Toast.makeText(this, "You can add geofences...",
Toast.LENGTH_SHORT).show();
            }
            else {
                Toast.makeText(this, "Background location access is necessary
for geofences" +
                        "to trigger...", Toast.LENGTH_SHORT).show();
            }
        }

    }

    @Override
    public void onMapLongClick(LatLng latLng) {
        if(Build.VERSION.SDK_INT >= 29){
            if(ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_BACKGROUND_LOCATION) ==
            PackageManager.PERMISSION_GRANTED){
                tryAddingGeofence(latLng);
            }
            else{
                if(ActivityCompat.shouldShowRequestPermissionRationale(this,
                        Manifest.permission.ACCESS_BACKGROUND_LOCATION)){
                    ActivityCompat.requestPermissions(this, new String[]
{Manifest.permission.ACCESS_BACKGROUND_LOCATION},
                            BACKGROUND_LOCATION_ACCESS_REQUEST_CODE);
```

```
                }
                else{
                    ActivityCompat.requestPermissions(this, new String[]
{Manifest.permission.ACCESS_BACKGROUND_LOCATION},
                            BACKGROUND_LOCATION_ACCESS_REQUEST_CODE);
                }
            }
        }
        else{
            tryAddingGeofence(latLng);
        }
    }

    private void tryAddingGeofence(LatLng latLng){
        mMap.clear();
        addMarker(latLng);
        addCircle(latLng, GEOFENCE_RADIUS);
        addGeofence(latLng, GEOFENCE_RADIUS);
    }


    @SuppressLint("MissingPermission")
    private void addGeofence(LatLng latLng, float radius){
        Geofence geofence = geofenceHelper.getGeofence(GEOFENCE_ID, latLng,
radius,
                Geofence.GEOFENCE_TRANSITION_ENTER |
Geofence.GEOFENCE_TRANSITION_DWELL | Geofence.GEOFENCE_TRANSITION_EXIT);
        GeofencingRequest geofencingRequest =
geofenceHelper.getGeofencingRequest(geofence);
        PendingIntent pendingIntent = geofenceHelper.getPendingIntent();

        geofencingClient.addGeofences(geofencingRequest, pendingIntent)
                .addOnSuccessListener(new OnSuccessListener<Void>() {
                    @Override
                    public void onSuccess(Void aVoid) {
                        Log.d(TAG, "onSuccess: Geofence Added...");
                    }
                })
                .addOnFailureListener(new OnFailureListener() {
                    @Override
                    public void onFailure(@NonNull Exception e) {
                        String errorMessage =
geofenceHelper.getErrorString(e);
                        Log.d(TAG, "onFailure: " + errorMessage);
                    }
                });
    }

    private void addMarker(LatLng latLng){
        MarkerOptions markerOptions = new MarkerOptions().position(latLng);
        mMap.addMarker(markerOptions);
    }

    private void addCircle(LatLng latLng, float radius){
        CircleOptions circleOptions = new CircleOptions();
        circleOptions.center(latLng);
```

```
        circleOptions.radius(radius);
        circleOptions.strokeColor(Color.argb(255,0,255,0));
        circleOptions.fillColor(Color.argb(64,0,255,0));
        circleOptions.strokeWidth(4);
        mMap.addCircle(circleOptions);
        System.out.println("privacy 2");
    }
}
```

**java/GeofenceBroadcastReceiver**

```
//Adapted from https://github.com/trulymittal/Geofencing
package com.example.geofenceapp;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.location.Location;
import android.util.Log;
import com.google.android.gms.location.Geofence;
import com.google.android.gms.location.GeofencingEvent;
import java.util.List;

public class GeofenceBroadcastReceiver extends BroadcastReceiver {
    private static final String TAG = "GeofenceBroadcastReceiv";
    private double person = 1;
    private double time = 1;
    private double loc = 0.2;
    private double newLat, newLong;
    final private double obfuscationfactor = 0.0003;

    @Override
    public void onReceive(Context context, Intent intent) {
        NotificationHelper notificationHelper = new
NotificationHelper(context);
        GeofencingEvent geofencingEvent = GeofencingEvent.fromIntent(intent);

        if(geofencingEvent.hasError()){
            Log.d(TAG, "onReceive: Error receiving geofence");
            return;
        }

        List<Geofence> geofenceList =
geofencingEvent.getTriggeringGeofences();
        for(Geofence geofence:geofenceList){
            Log.d(TAG, "onReceive TOSTRING: " + geofence.toString());
            Log.d(TAG, "onReceive: REQUESTID" + geofence.getRequestId());
        }

        Location location = geofencingEvent.getTriggeringLocation();
        int transitionType = geofencingEvent.getGeofenceTransition();
```

```
        double finalprivacyvalue = person * time * loc;
        double rand = finalprivacyvalue * obfuscationfactor;

        if(Math.random() * 1 < 0.5)
            newLat = location.getLatitude() + Math.random() * (rand - 0) + 0;
        else
            newLat = location.getLatitude() - Math.random() * (rand - 0) + 0;

        if(Math.random() * 1 < 0.5)
            newLong = location.getLongitude() + Math.random() * (rand - 0) +
0;
        else
            newLong = location.getLongitude() - Math.random() * (rand - 0) +
0;

        System.out.println("Old Coordinates: " + location.getLatitude() + ""
+ location.getLongitude());
        System.out.println("New Coordinates: " + newLat + "" + newLong);

        switch (transitionType) {
            case Geofence.GEOFENCE_TRANSITION_ENTER:
                break;

            case Geofence.GEOFENCE_TRANSITION_DWELL:
                notificationHelper.sendHighPriorityNotification("Privacy
Coordinate", newLat + ", " + newLong, MapsActivity.class);
                break;

            case Geofence.GEOFENCE_TRANSITION_EXIT:
                break;
        }
    }
}
```

**java/GeofenceHelper**

```
//Adapted from https://github.com/trulymittal/Geofencing
package com.example.geofenceapp;

import android.app.PendingIntent;
import android.content.Context;
import android.content.ContextWrapper;
import android.content.Intent;
import com.google.android.gms.common.api.ApiException;
import com.google.android.gms.location.*;
import com.google.android.gms.location.Geofence;
import com.google.android.gms.location.GeofencingRequest;
import com.google.android.gms.maps.model.LatLng;

public class GeofenceHelper extends ContextWrapper {
    private static final String TAG = "GeofenceHelper";
    PendingIntent pendingIntent;

    public GeofenceHelper(Context base) {
        super(base);
```

```
    }

    public GeofencingRequest getGeofencingRequest(Geofence geofence){

        return new GeofencingRequest.Builder()
                .addGeofence(geofence)
                .setInitialTrigger(GeofencingRequest.INITIAL_TRIGGER_ENTER)
                .build();
    }

    public Geofence getGeofence(String ID, LatLng latLng, float radius, int
transitionTypes){
        return new Geofence.Builder()
                .setCircularRegion(latLng.latitude, latLng.longitude, radius)
                .setRequestId(ID)
                .setTransitionTypes(transitionTypes)
                .setLoiteringDelay(5000)
                .setExpirationDuration(Geofence.NEVER_EXPIRE)
                .build();
    }

    public PendingIntent getPendingIntent(){
        if(pendingIntent != null){
            return pendingIntent;
        }
        Intent intent = new Intent(this, GeofenceBroadcastReceiver.class);
        pendingIntent = PendingIntent.getBroadcast(this, 2607, intent,
                PendingIntent.FLAG_UPDATE_CURRENT);
        return pendingIntent;
    }

    public String getErrorString (Exception e){
        System.out.println("METHOD GETERROR STRING GEOFENCEHELPER");
        if(e instanceof ApiException){
            ApiException apiException= (ApiException) e;
            switch (apiException.getStatusCode()){
                case GeofenceStatusCodes
                        .GEOFENCE_NOT_AVAILABLE:
                    return "GEOFENCE_NOT_AVAILABLE";
                case GeofenceStatusCodes
                        .GEOFENCE_TOO_MANY_GEOFENCES:
                    return "GEOFENCE_TOO_MANY_GEOFENCES";
                case GeofenceStatusCodes
                        .GEOFENCE_TOO_MANY_PENDING_INTENTS:
                    return "GEOFENCE_TOO_MANY_PENDING_INTENTS";
            }
        }
        return e.getLocalizedMessage();
    }
}
```

**java/NotificationHelper**

```
//Adapted from https://github.com/trulymittal/Notifications.git

package com.example.geofenceapp;

import android.app.Notification;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.ContextWrapper;
import android.content.Intent;
import android.graphics.Color;
import android.os.Build;
import androidx.annotation.RequiresApi;
import androidx.core.app.NotificationCompat;
import androidx.core.app.NotificationManagerCompat;

import java.util.Random;

public class NotificationHelper extends ContextWrapper {
    private static final String TAG = "NotificationHelper";

    public NotificationHelper(Context base) {
        super(base);
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            createChannels();
        }
    }

    private String CHANNEL_NAME = "High priority channel";
    private String CHANNEL_ID = "com.example.notifications" + CHANNEL_NAME;

    @RequiresApi(api = Build.VERSION_CODES.O)
    private void createChannels() {
        NotificationChannel notificationChannel = new
NotificationChannel(CHANNEL_ID, CHANNEL_NAME,
NotificationManager.IMPORTANCE_HIGH);
        notificationChannel.enableLights(true);
        notificationChannel.enableVibration(true);
        notificationChannel.setDescription("this is the description of the
channel.");
        notificationChannel.setLightColor(Color.RED);

notificationChannel.setLockscreenVisibility(Notification.VISIBILITY_PUBLIC);
        NotificationManager manager = (NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);
        manager.createNotificationChannel(notificationChannel);
    }

    public void sendHighPriorityNotification(String title, String body, Class
activityName) {
```

```
        Intent intent = new Intent(this, activityName);
        PendingIntent pendingIntent = PendingIntent.getActivity(this, 267,
intent, PendingIntent.FLAG_UPDATE_CURRENT);

        Notification notification = new NotificationCompat.Builder(this,
CHANNEL_ID)
                .setSmallIcon(R.drawable.ic_launcher_background)
                .setPriority(NotificationCompat.PRIORITY_HIGH)
                .setStyle(new
NotificationCompat.BigTextStyle().setSummaryText("summary").setBigContentTitl
e(title).bigText(body))
                .setContentIntent(pendingIntent)
                .setAutoCancel(true)
                .build();
        NotificationManagerCompat.from(this).notify(new Random().nextInt(),
notification);
    }
}
```

## BIBLIOGRAPHY

[1] Abul, O., & Bayrak, C. (2018). From location to location pattern privacy in location-based services. *Knowledge and Information Systems*, *56*(3), 533–557. https://doi.org/10.1007/s10115-017-1146-x

[2] Alam, T., Hadi, A. A., & Najam, R. Q. (2021). Designing and implementing the people tracking system in the crowded environment using mobile application for Smart Cities. *International Journal of System Assurance Engineering and Management*, *13*(1), 11–33. https://doi.org/10.1007/s13198-021-01277-7

[3] Ardagna, C., Cremonini, M., De Capitani di Vimercati, S., & Samarati, P. (2011). An obfuscation-based approach for protecting location privacy. *IEEE Transactions on Dependable and Secure Computing*, *8*(1), 13–27. https://doi.org/10.1109/tdsc.2009.25

[4] Bunn, J. (2021, February 2). *What are location-based services?* Ancoris. Retrieved March 20, 2022, from https://www.ancoris.com/blog/what-are-location-based-services

[5] Freedman, M. (2021, November 10). *What are location-based services?* Business News Daily. Retrieved March 18, 2022, from https://www.businessnewsdaily.com/5386-location-based-services.html

[6] Jiang, H., Li, J., Zhao, P., Zeng, F., Xiao, Z., & Iyengar, A. (2022). Location privacy-preserving mechanisms in location-based services. *ACM Computing Surveys*, *54*(1), 1–36. https://doi.org/10.1145/3423165

[7] Kemmis, A. (n.d.). *What is geofencing? everything you need to know about location-based marketing*. Inbound Marketing Agency. Retrieved February 16, 2022, from https://www.smartbugmedia.com/blog/what-is-geofencing

[8] *Location-based marketing with geofencing*. Csek Creative - New Site. (n.d.). Retrieved March 18, 2022, from https://www.csekcreative.com/location-based-marketing-with-geofencing/

[9] Memon, I. (2015). Authentication user's privacy: An integrating location privacy protection algorithm for secure moving objects in location based services. *Wireless Personal Communications*, *82*(3), 1585–1600. https://doi.org/10.1007/s11277-015-2300-y

[10] Namiot, D., & Sneps-Sneppe, M. (2013). Geofence and Network Proximity. *Internet of Things, Smart Spaces, and Next Generation Networking*, 117–127. https://doi.org/10.1007/978-3-642-40316-3_11

[11] Shokri, R., Theodorakopoulos, G., & Troncoso, C. (2017). Privacy games along location traces. *ACM Transactions on Privacy and Security*, *19*(4), 1–31. https://doi.org/10.1145/3009908

[12] Sun, G., Chang, V., Ramachandran, M., Sun, Z., Li, G., Yu, H., & Liao, D. (2017). Efficient location privacy algorithm for internet of things (IOT) services and applications. *Journal of Network and Computer Applications*, *89*, 3–13. https://doi.org/10.1016/j.jnca.2016.10.011

[13] *What is Geofencing*. BuildFire. (n.d.). Retrieved February 2022, from https://dl-acm-org.ezaccess.libraries.psu.edu/doi/10.1145/3423165

[14] *What is geospatial data?* IBM. (n.d.). Retrieved March 18, 2022, from https://www.ibm.com/topics/geospatial-data

[15] *What is location-based marketing*. Marketing Evolution. (n.d.). Retrieved March 18, 2022, from https://www.marketingevolution.com/knowledge-center/topic/marketing-essentials/location-based-marketing

# ACADEMIC VITA

# Eileen Pi

## EDUCATION

**The Pennsylvania State University |** University Park, PA
Schreyer Honors College
B.S. in Cybersecurity Analytics and Operations (Geopolitics), May 2022
Minor in Psychology, May 2022
Dean's List: Fall 2018-Fall 2021

## EXPERIENCE

**Cyber Scholarship Program Scholar (CySP) | U.S. Department of Defense,** May 2020-June 2022

- Highly selective recruitment program of the nation's top cyber talent and the retention of DoD personnel who have skills necessary to meet DoD's cyber requirements and help secure our nation against threats of information systems and networks

**Computer Scientist | U.S. Cyber Command | U.S. Department of Defense,** May 2021-August 2021

- Conducted Department of Defense (DoD) cybersecurity operation administration and technical implementation for Persistent Cyber Training Environment (PCTE)
- Applied and integrated DoD organizational standards and NIST security policies with the focus on policy documentation of the Risk Management Framework
- Created organizational chart for PCTE and RACI chart to clearly mapped out organizational functions, roles, and responsibilities to facilitate PCTE operations.
- Conducted cybersecurity framework review to identify gaps and weakness in current platforms and maximize application safeguards to harden network security
- Identify and track key system assets with a focus on exploit impact analysis
- Formulate potential exploit tactics and techniques by national actors or criminals to compromises key system assets with a focus on the Risk Management Framework
- Researched the joint planning process and operation design as well as cyberspace operations

**Information Technology Intern | Morey's Piers, Beachfront Water Parks & Resorts**, June-August 2019

- Conducted installation, operation, and maintenance of wireless and wired networks for Point of Service systems and credit card authorization and payment systems
- Conducted help desk operations, emergency hardware and software support, system status monitoring, and commercial data mapping
- Conducted network administration, system installation and updates, and process development
- Provide effective communication for end-user training and writing of user instructions and system manuals

**Learning Assistant | Penn State College of Information Sciences and Technology**, January 2019-May 2022

- LA for Computer Systems Literacy (CYBER100), Introduction to Application Programming (IST 140), Organization of Data (IST210), and Security and Risk Management (IST 456)
- Conducted in-class setup and instructional activities, instructed specialized software, supervised virtual labs, provided feedback on writing, out-of-class one-on-one tutoring, and other learning-focused responsibilities

**IST Diplomat | Penn State College of Information Sciences and Technology,** April 2020-May 2022

- Acted as student diplomat representative of the IST
- Conducted recruitment and retention of prospective and current students for the selective IST programs

**Auxiliary Officer | Penn State University Police and Public Safety**, September 2018-June 2019

- Conducted public safety and law enforcement activities to include traffic enforcement and safety operations for the University Police at PSU sports gathering, special events, and public community activities

## VOLUNTEER WORK

**Medical Volunteer | Love Volunteers,** July-August 2016 and June-July 2017
Ngora, Uganda and Quito, Ecuador

- Worked alongside local doctors and nurses in screening for HIV and Hepatitis B, assisted deliveries and maternal care, prepared prescriptions, and laboratory examinations to urban and rural underprivileged poor indigent population

## SKILLS

- Software Proficiency
  - Cybersecurity: Wireshark, Metasploit, Kali Linux PenTesting, Splunk, FTK Imager, Autopsy, Volatility
  - Computer programming: Java, JavaScript, SQL, HTML
  - Operating Systems and Software: Microsoft Windows, MacOS, Linux, Kali Linux, iOS, Ubuntu
  - Multimedia: Adobe Photoshop, Adobe Lightroom, iMovie, Audacity
- Language Skills: Fluent English; Intermediate Chinese Mandarin; Elementary Spanish

## LEADERSHIP AND ACTIVITIES

**President | Asian American Christian Fellowship**, April 2020-May 2022

- Lead weekly organization meetings, facilitate conversation, liaison between students and faculty
- Represent organization with Penn State student affairs and other campus organizations
- Manage organization's financial records via Excel and Google Drive

**Secretary | Penn State Business and Society House**, January 2019-September 2020

- Coordinate with the Chair and Housing coordinators to plan and conduct meetings and circulate reports
- Create design and update campaigns and website for society's communication purposes

**Freelance Photographer**, 2012–present

- Document (capture, edit, and produce black/white and digital photographs) events and people using digital imagery, graphic processing, and video enhancement, design, and productions