VEHICLE PSEUDONYM ASSOCIATION ATTACK MODEL

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Computer Science

by

Pierson Yieh

June 2018

COMMITTEE MEMBERSHIP

TITLE:	Vehicle Pseudonym Association Attack Model
AUTHOR:	Pierson Yieh
DATE SUBMITTED:	June 2018
COMMITTEE CHAIR:	Bruce DeBruhl, Ph.D. Professor of Computer Science
COMMITTEE MEMBER:	Theresa Migler-VonDollen, Ph.D. Professor of Computer Science
COMMITTEE MEMBER:	Lubomir Stanchev, Ph.D. Professor of Computer Science
COMMITTEE MEMBER:	Zachary Peterson, Ph.D. Professor of Computer Science

ABSTRACT

Vehicle Pseudonym Association Attack Model

Pierson Yieh

With recent advances in technology, Vehicular ad hoc Networks (VANETs) have grown in application. One of these areas of application is Vehicle Safety Communication (VSC) technology. VSC technology allows for vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications that enhance vehicle safety and driving experience. However, these newly developing technologies bring with them a concern for the vehicular privacy of drivers. Vehicles already employ the use of pseudonyms, unique identifiers used with signal messages for a limited period of time, to prevent long term tracking. But can attackers still attack vehicular privacy even when vehicles employ a pseudonym change strategy? The major contribution of this paper is a new attack model that uses long-distance pseudonym changing and short-distance non-changing protocols to associate vehicles with their respective pseudonyms.

ACKNOWLEDGMENTS

Thanks to:

- Dr. Bruce DeBruhl for his guidance during this project.
- Sean Bayley, Atticus Liu, and Aaron Otis for their contributions to this project.
- Andrew Guenther, for uploading the LaTeXtemplate

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

With recent technological advances in intelligent transportation systems (ITS), Vehicular Ad-Hoc Networks (VANETs) have seen expanded applications. VANETs are systems of vehicles communicating with each other and roadside infrastructures. One application for VANETs is Vehicle Safety Communication (VSC), which aim to enhance vehicle safety and the driving experience. VSC can be further broken down into vehicle-to-vehicle (V2V) communications and vehicle-to-infrastructure (V2I) communications. An example of V2V communication is platooning, where vehicles closely follow other vehicles aided by wirelessly exchanging steering and acceleration information. Examples of V2I communication are automatic tollbooth payments, where vehicles can automatically send their occupants' payment information to tollbooths without having to stop, and emergency roadside warnings, where roadside infrastructures can broadcast to vehicles information about upcoming dangerous weather and road conditions [22]. Because VANETs use wireless technology for communication, an attacker, an agent outside of the network, can easily pick up signal packets with proper equipment [4, 13]. This brings into question whether a malicious attacker can track users for an extended period of time using these messages, in what we refer to as a tracking attack. A tracking attack would undermine vehicular security and be an infringement upon user privacy.

One means of mitigating a tracking attack is using pseudonyms in VANET messages [12]. Pseudonyms are temporary unique identifiers used by vehicles when sending VSC messages that are switched regularly to prevent tracking over long periods of time. Pseudonyms are distributed to vehicles by a trusted third-party Certificate Authority (CA), with vehicles' owners being associated with their vehicles'

pseudonyms, for liability reasons [20]. Requiring a CA instead of simply allowing vehicles to generate their own pseudonyms ensures authenticity, prevents pseudonym spoofing, and mitigates pseudonym collisions. Pseudonym spoofing is when a node alters its pseudonym to pose as another user. Pseudonym collisions are when two vehicles have the same pseudonym. While a simple strategy for protecting user privacy in VANETS is to frequently switch pseudonyms, this is not an ideal solution because pseudonym changes are expensive [16]. The cost for changing pseudonyms comes from the limited number of pseudonyms a vehicle can store and the expense, or impossibility, of downloading new pseudonyms. To most effectively change pseudonyms, research has been done into pseudonym change strategies, or algorithms that determine when to best change pseudonyms. I discuss a number of pseudonym change strategies in Chapter 2. While pseudonym change strategies aim to maximize the utility of each pseudonym change, they do not guarantee preserving user privacy.

While previous attacks have been devised to test the effectiveness of user privacy enhancing technologies, they have focused only on single radio systems, systems that assume the usage of or are only concerned with a single radio protocol. I propose that a malicious attacker can develop a more effective attack by combining multiple attack vectors leveraging available information that current attacks do not consider. Previous works have all assumed a single radio, but with modern technology, many vehicles are equipped with many radios that are used for different purposes. Therefore, in this work, I develop a generalizable privacy attack that targets two common radio protocols used by vehicles in VANETs, long-distance pseudonym changing protocol and short-distance non-changing protocol. This attack can be used by a passive adversary to attack vehicular privacy. In my attack, I deanonymize vehicles by associating signals' pseudonyms to each other, that is to accurately predict when two signals originate from the same vehicle. This information is then used to recreate a vehicle's path, thus tracking the vehicle over an extended period of time.

I use the general long-distance pseudonym changing and short-distance non-changing protocol types because of their common analogies in commodity systems. Examples of long-distance pseudonym changing radio protocols are Dedicated Short Range Communication (DSRC) and 5G, and examples of short-distance non-changing signal protocols are WiFi, Bluetooth, Zigbee, and tire-pressure monitoring systems. I use DSRC and WiFi in my experiments due to their widespread usage in modern vehicles, but the attack can be altered to use any signals of the two types. Using both radio protocols allows me to uniquely identify vehicles as well as expand my area of knowledge of the identified vehicles. I use the short-distance non-changing signal protocol to uniquely identify vehicles within my attack area, and I use the long-distance pseudonym changing protocol due to its greater range and area of effect to expand my knowledge of vehicles' locations once I have associated signals to each other.

I use SUMO (Simulation of Urban Mobility) [2], an open source simulation package commonly used to simulate realistic road traffic for academic purposes, to simulate vehicle traffic in three real-world maps. I then place listeners throughout the maps to simulate an attacker placing real listeners to eavesdrop on signal packets. My listeners gather signal packets based on the SUMO vehicle simulations, from which I attack vehicular privacy by associating identifiers and pseudonyms seen by my listeners. I attack a number of realistic vehicle traffic simulations and present my findings in Chapter 5. For WiFi-to-DSRC associations, my attack model achieved around an 80% average precision and 20% average recall per vehicle. For DSRC-to-DSRC associations, my attack model consistently achieved at least a 96% precision and between 18% to 38% recall depending on the map.

1.1 Social and Ethical Implications

The social implications of vehicle tracking is a debate of weighing economical benefits and safety versus privacy. One application of vehicle tracking is as evidence in court cases and insurance claims. With vehicle tracking, law enforcement can more accurately track suspicious vehicles without having to spend the extensive resources that are required in previously conventional means of tracking such as stakeouts, collecting traffic camera footage, and tailing vehicles. Tracking vehicles can also help resolve insurance claims. If there is a log of a vehicle's exact location at any given time, one can figure out definitively which party is at fault in what would otherwise be an ambiguous car accident.

While VANET messages can be used to help law enforcement in criminal cases and insurance liability claims, a malicious adversary with the proper equipment can attempt the same tracking. These forms of tracking violate the contextual integrity of VANET messages. Contextual integrity is a theory of privacy proposed by Helen Nissenbaum that defines privacy as the appropriate flow of information based on contextual norms [14, 15]. In contextual integrity, norms of appropriateness dictate what information about a person is appropriate, or fitting, to reveal in a particular context. Norms of distribution dictate how that information is shared based on particular context. While it can be argued that the ability to track vehicles is already available through the use of publicly available cameras, the majority of people are not actively concerned with the cameras' presence due to the services they provide [25] and the belief that they, the individual, is not of enough particular interest to be tracked [14]. However, the ability of a malicious attacker to track a vehicle and its passengers over an extended period of time is a violation of the norms of distribution and appropriateness, as most people do not expect to be tracked. It can also be argued that, when users willingly participate in and use VANET technology, they

are entrusting their information to those that develop the technology. Not effectively safeguarding this information or defending against attacks that undermine their users' privacy would be a breach of this trust by the developers with their users.

Even with regards of legal monitoring, there is what is known as the "chilling effect." The chilling effect is when people do less of something, even if it is a normal, legal activity, due to the fear of being monitored, regardless of if they are actually being monitored or not. Simply the potential for a tracking attack can be an infringement upon the freedom of people to openly travel and partake in activities.

1.2 Basic Attack Model

For my attack, I assume the role of a passive attacker, as opposed to an online or real-time attacker. Passive attackers collect data, then attack the system afterwards. Online or real-time attackers are able to attack the system as they are collecting data. I gather signal packets from listeners that have been placed throughout the area of observation, working under the assumption that the only information I can gather is the identifier or pseudonym associated with a signal message, not any of the message content or location data. I then try to associate the identifiers and pseudonyms that I have seen to those that originate from the same vehicle by constructing two association matrices. I assume easy access to the message pseudonyms due to safety messages being unencrypted because they do not contain any sensitive information[19]. Safety messages also often contain vital information that can prevent immediate threats such as collision avoidance. Therefore, the overhead of establishing a secure protocol between communicating vehicles is often considered too expensive.

The first matrix associates pseudonyms to other pseudonyms. I conclude that for any pair of pseudonyms they are disassociated, associated, possibly associated, or that I do not have enough information to draw a conclusion about the relationship between

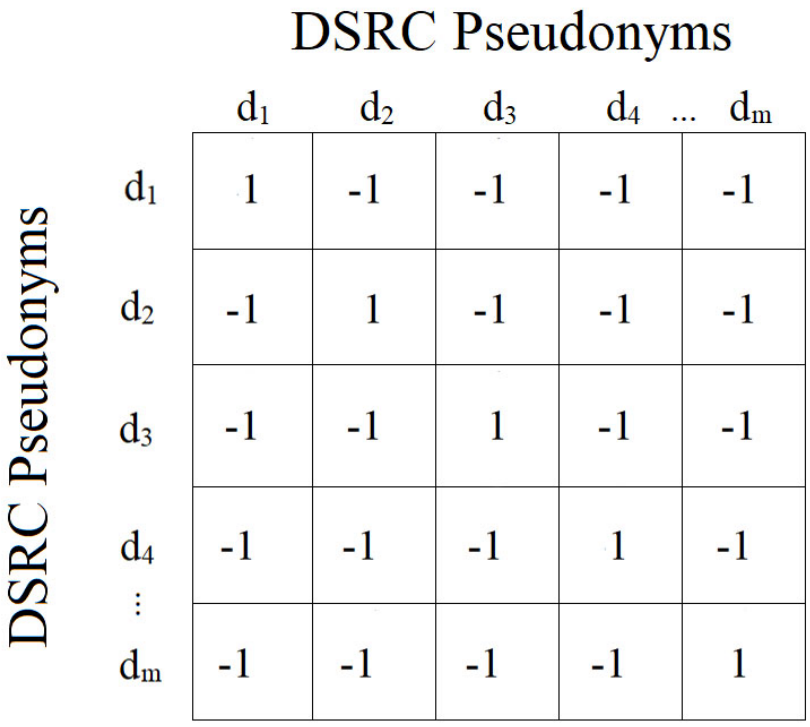


Figure 1.1: Initial D2D Matrix

the pair. If two pseudonyms are seen in different locations at the same point in time, then I conclude that they are disassociated. If a pseudonym appears immediately after another pseudonym disappears, I hypothesize that a vehicle changed its pseudonym while in my area of observation, and therefore the two pseudonyms are associated. Next, I try to match enter events to their corresponding exit events, where enter and exit events are vehicles entering or exiting my area of observation, respectively, to try and associate pseudonyms. Finally, I apply the transitive property across the matrix, extending disassociations and associations, but not possible associations. Figure 1.1 illustrates the initialization of the D2D matrix.

The second matrix contains the probabilities between identifiers to pseudonyms associations. The matrix is a stochastic matrix where each element contains the

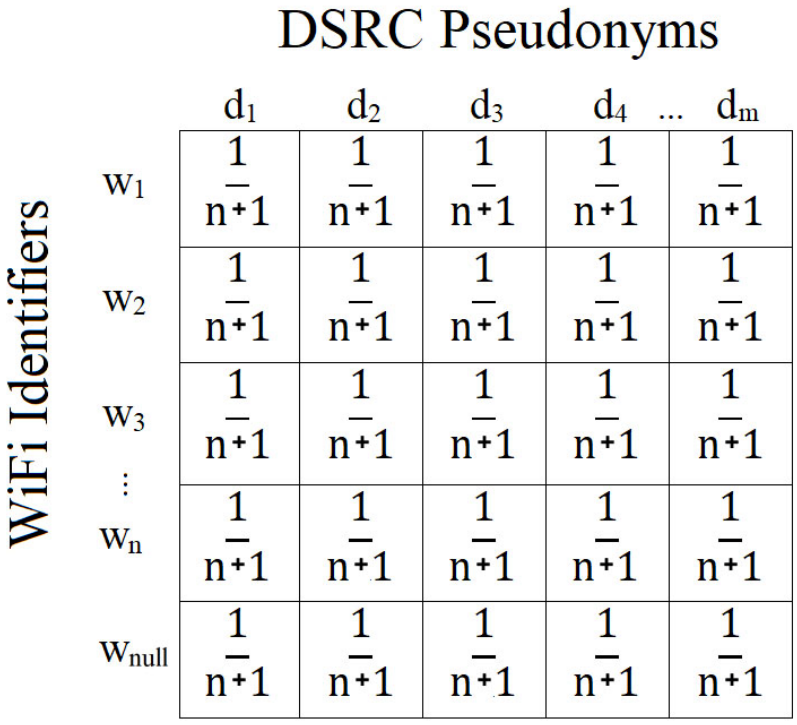


Figure 1.2: Initial W2D Matrix

probability that the identifier and the pseudonym that its row and column correspond to, respectively, are associated. The matrix is also a left stochastic matrix, meaning that each column sums to one because they represent pseudonyms. Since identifiers are non-changing, and each vehicle can only have one identifier and pseudonyms are changing, an identifier can be associated with many pseudonyms, while a pseudonym can only be associated with a single identifier. That is why columns, which represent pseudonyms, sum to one. I initialize each element to be $\frac{1}{N+1}$ where N is the number of identifiers observed, that is each pseudonym is initially equally likely to be associated to any identifier. Figure 1.2 illustrates the initialization of the W2D matrix.

I then find disassociation using the same method of finding disassociations that is used in the first matrix construction. An identifier and a pseudonym are found to

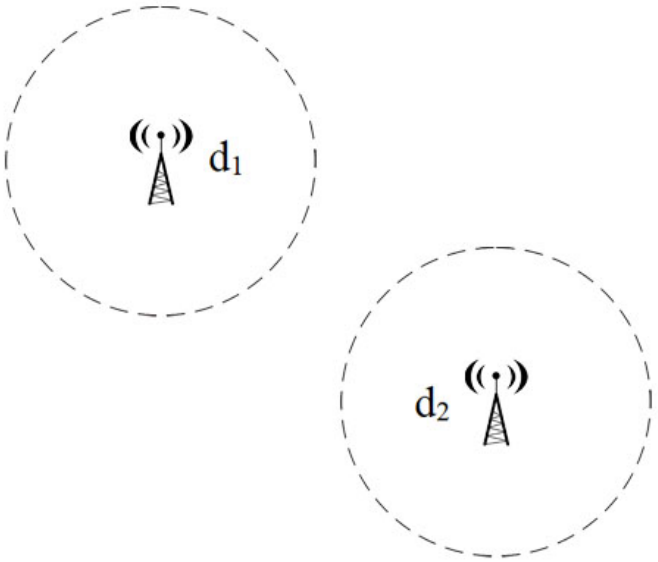


Figure 1.3: Example Disassociation

be disassociated if they are seen by different listeners at the same point in time. An example of this is illustrated in Figure 1.3. I set the probability of the disassociated identifier-pseudonym pair, that is, the element in the matrix representing the pair, to be zero, and distributing its original value among the other elements in that row, thus increasing those probabilities and maintaining the column sum of one.

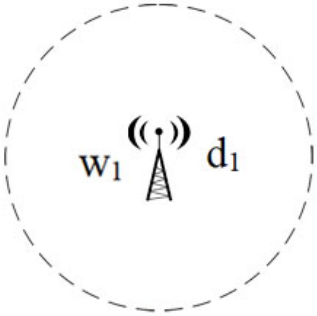


Figure 1.4: Example Known Association

Next, I find known associations, that is, associations that are so likely to be correct, that I consider them to be correct. If an identifier and a pseudonym are seen

together at the same listener, at the same time, and no other pseudonyms are around, then the pair is considered to be associated. I set their probability to one, as well as all other probabilities in that column to be zero. If an identifier and a pseudonym are seen by the same listener and at the same time, then I say that they are seen together or that one is seen with the other.

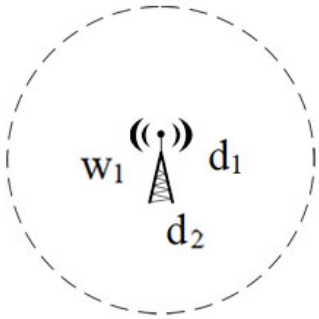


Figure 1.5: Example Possible Associations

I then find possible associations, that is, associations that are not guaranteed to be correct as known associations, but still likely. There are three levels of possible associations, with the first level being the most likely associations, and subsequent levels being less likely. The first level of possible associations is between pairs of identifiers and pseudonyms where the identifier is seen with the pseudonym, but other pseudonyms are also present at that listener; an example of this is illustrated in Figure 1.5. The second level of possible association between an identifier and a pseudonym is when the identifier is seen with a pseudonym that is associated with the pseudonym in question. The third level of possible association between an identifier and a pseudonym is when the identifier is seen with a pseudonym that is possibly associated with the pseudonym in question. The second and third level associations use pseudonym-to-pseudonym associations from the previously constructed pseudonym-to-pseudonym matrix.

I also try to find pseudonyms originating from vehicles that do not support the short-distance non-changing protocol. I do this on the proposition that a pseudonym seen without any overlapping identifier is more likely to originate from a vehicle without a short-distance non-changing protocol than a pseudonym that is seen with an overlapping identifier.

Finally, I use a confidence threshold value to determine which associations I am considering to be actual associations. Association with a probability greater than the confidence threshold I deem to be actual associations. The found associations are then used to recreate vehicles' paths by drawing a Euclidian path between the listeners where the vehicle was seen at.

1.3 Contributions

To summarize, I make the following contributions in this thesis:

- An attack model that leverages two radio protocols to deanonmiyze vehicles by associating unique identifiers with temporaroy pseudonyms and temporary pseudonyms with other temporary pseudonyms, regardless of radio market penetration rates.
- A Euclidian path reconstruction method using associations found by the attack model.
- Tunable and configurable parameters that allow the attack model to be applicable across different scenarios.
- A more effective weight calculation than previous research when constructing the weighted bipartite graph in the Exit-to-Enter Attack, as well as addition of a maximum weight threshold to filter out less likely associations.

Chapter 2

RELATED WORKS

This project expands on previous works that investigated the possibility and effectiveness of tracking wireless devices and vehicles using the beacon messages transmitted periodically by the tracking targets. Recent works showed that an attacker can track many mobile devices with accuracy comparable to GPS and provide high-accuracy trajectory information by simply using inexpensive off-the-shelf equipment [4, 13]. Furthermore, it has been suggested that vehicles' tire pressure monitoring systems (TPMS) contain vulnerabilities that would allow an attacker to perform similar tracking attacks on vehicles [8]. This is of particular interest to me because my project involves using existing built-in vehicle technology protocols to perform tracking attacks.

2.1 Pseudonym Change Strategies

Because of the concern of vehicle tracking through the use of V2X protocols, there have been a number of works exploring effective means of defending against such attacks. A key means of defending against vehicle tracking is the use of pseudonyms [17]. A pseudonym is a unique identifier associated with a signal emitted by a vehicle to allow unique identification for a period of time, as some signal usages require identification, but also changed periodically to prevent long-term tracking. The simple strategy for protecting privacy in VANETs by frequently changing pseudonym is not ideal because pseudonym changes are expensive due to limited vehicle pseudonym storage capacities, limited download capabilities for new pseudonyms, and increased costs on various applications [17, 20]. Vehicles are equipped with a set of pseudonyms

distributed by a trusted Certificate Authority, and either cycle through and re-use pseudonyms or have to acquire new pseudonyms from the CA [11]. Therefore, effective pseudonym change strategies, algorithms that dictate when a vehicle changes pseudonyms, have been developed to maximize the effectiveness of each pseudonym change, while minimizing the number of changes required.

My experiments use traffic simulations where vehicles employ a periodic change strategy, in which the signal protocols dictate how often a vehicle broadcasts a message (heart-rate) and changes its pseudonym (change-rate), with all vehicles having the same heart-rate and change-rate if they are using the same protocol. This change strategy is simple and does not require cooperation among neighboring vehicles to coordinate pseudonym changes, which prevents malicious adversaries to disrupt privacy gains of pseudonym changes by vehicles within the VANET, a topic which has been investigated in graduate student Nicholas Plewtong's thesis paper [18].

Another pseudonym change strategy is a synchronous change strategy [12]. The synchronous change strategy can be classified as a type of a position change strategy, where vehicles change pseudonyms when a minimum threshold of vehicle density within their proximity is met. The synchronous change strategy requires coordination among the vehicles in the VANET to maximize effectiveness of pseudonym changes. Vehicles that change pseudonyms together would prove to be more difficult for an attacker to associate their new pseudonyms to their respective old pseudonyms due to the lack of distinguishing features and increased number of possible association pairs. The synchronous change strategy has vehicles ready to change pseudonyms c_{min} seconds after their last change. Vehicles indicate their readiness to change by setting a *change* flag within their broadcast messages. A change is triggered when there are k – 1 other vehicles with their *change* flag set within the transmission range. This allows for a synchronous pseudonym change by all vehicles ready to change pseudonyms within that given area. If the threshold of k – 1 vehicles is not met within c_{max}

seconds after the last change, then the vehicle changes pseudonyms anyways. This is to prevent vehicles from perpetually waiting for the number of vehicles threshold to be met in sparse areas or in areas where this strategy has not been adopted [12]. On its own without the necessary modifications, the synchronous change strategy would be susceptible to malicious adversaries posing as compliant vehicles nodes that aim to disrupt the vehicular privacy of nodes within the VANET [18]. A similar change strategy called the density-based location privacy scheme has also been proposed that uses the same concept of changing pseudonyms when enough other vehicles are within proximity to make old-new pseudonym associations more difficult [23].

Another type of position change algorithm is a similar status algorithm. In a similar status algorithm, vehicles coordinate pseudonym changes by looking for other vehicles with a similar status as their own, with a pseudonym change occurring when a given number of vehicles with a similar status are within proximity of a respective vehicle. This strategy allows for many different things to be considered features of a vehicle's status such as speed, direction, and number of neighbors [7]. However, this strategy requires the broadcasting of additional descriptive information about the vehicle, which can be used by adversaries to further distinguish vehicles from each other [5].

AMOEBA is another means of defending user privacy that takes advantage of the clustering nature VANETs to prevent malicious attackers [21]. While not a pseudonym change strategy, *AMOEBA* is a privacy scheme that employs the use of pseudonyms, namely a single group pseudonym representative of all vehicles within a given cluster. Groups are formed by vehicles that move with a similar velocity and relative to each other, and have a fully connected network graph within them to allow for communication among their respective members. A leader of the group is elected at random and broadcasts on behalf of the group members, while other members can remain silent. *AMOEBA* also defines the use of silent periods to prevent linka-

bility between two locatable broadcasts. Random silent periods are used to prevent trackability. An example would be if a vehicle changes its pseudonym from A to A'. Initially entering a network and broadcasting as A, after having changed and waited a random silent period, it begins broadcasting as A'. If another vehicle had changed pseudonyms from B to B' within that silent period, an attacker may be misled to tracking the neighboring vehicle.

2.2 Previous Attack Models

A common attack used to quantitatively compare pseudonym and tracking related defenses is trying to match enter events with their respective exit events [6, 3]. Enter and exit events represent when a vehicle enters or exits a mix zone, respectively. A mix zone is an area outside of an attacker's area of observation, where vehicles can become mixed together without an attacker's knowledge.

The premise of this attack is to match corresponding enter and exit events using previously learned data. The attack is broken into two phases, a learning phase and an attack phase. During the learning phase, the attacker records the number of vehicles that travel between two areas of observation and the average time each vehicle took to make that respective trip. The data learned is limited to what can be learned from observing pseudonyms for the pseudonym's given lifespan, that is, the time before the vehicle changes pseudonyms. This means that due to vehicles changing pseudonyms, the number of vehicles recorded to have traveled between two locations can be greater than the actual number of vehicles. During the attack phase, the attack first actively attempts an online attack to match each newly observed exit event to a previously observed enter event. This step simply matches events if they broadcast the same pseudonyms and removes the events from the set of events that are used later. After, all unmatched events are used to create a bipartite graph, with the two distinct sets

being exit and enter events. Edges are assigned between an exit and an enter event with a weight equal to the number of vehicles that traveled between the two points where the events were observed divided by the average time each vehicle took. A penalty is added the farther the actual trip time was from the average trip time. If during the learning phase, no trip was recorded between the two locations, then a small weight of 0.1 is given. When the bipartite graph is complete, the solution is a matter of solving the linear sum problem and getting a minimal cost perfect match of the graph. The success of the attack is measured by how many pairs of events were correctly matched.

I use this attack as the basis for one source of information gathering when constructing my pseudonym-to-pseudonym association matrix. Note that their definitions of exit and enter events are the reverse of my usage of the terms in later chapters when discussing my attack model. I define enter and exit events as when a vehicle enters or exits my area of observation (i.e., exits or enters the mix zone), as opposed to when a vehicle enters or exits the mix zone. I go into more detail about the improvements I add to this attack and how I use the gathered information in Chapter 3 and Chapter 4.

Attacks have also been developed that take advantage of specific message protocol features. In *Examining Privacy in Vehicular Ad-Hoc Networks*, the authors break down the privacy vulnerabilities of the DSRC protocol stack [5]. Within SAE J2735, the standard message structure for DSRC messages, a vehicle is identified by a 4-byte temporary identifier pseudonym, while also broadcasting information such as the vehicle's GPS coordinates, motion information, and vehicle size that can be seen by anyone. The authors argue that DSRC messages leak enough information that an attacker can circumvent the temporary nature of the pseudonym and track vehicles despite switching pseudonyms, as well as linking pseudonyms to the actual vehicle or owner. The authors claim that attackers can use statistical methods, similar to

my attack, to track vehicles regardless of pseudonym switches. The main means of doing so is when a vehicle switches pseudonyms, an attacker will see a pseudonym no longer transmitting, and a new pseudonym begins transmitting within close proximity. The effectiveness of this attack increases with greater coverage and can also take advantage of the descriptive information contained in DSRC messages to better associate pseudonyms in the case of multiple vehicles simultaneously switching pseudonyms. I use a similar method in one of my matrix constructions, but I am limited to the pseudonym information of a new pseudonym appearing shortly after an old pseudonym disappears. The authors then claim that by using this location information, attackers can link pseudonyms back to the vehicle owners. Aside from the direct linkage of pseudonyms to their owners by gaining access to the pseudonym database, an attacker can use the location information to build a profile for certain vehicles. Knowing where vehicles stop and go at what times, an attacker can correlate points of interest to buildings and times visited to possibly discover where the user works and lives. After narrowing down buildings, an attacker can further pinpoint the user by performing a lookup of owners and occupants of the buildings for residency and employee directories for businesses in that area.

The authors describe a more general version of the previously mentioned synchronous pseudonym strategy [12] as an effective pseudonym change strategy to prevent tracking, although the use of a similar status algorithm [7] would likely be even more effective, as DSRC messages already contain descriptive information about a vehicle. They also cite the use of a group pseudonym as a means of defending privacy [21], though they do note that an attacker can attack the point when a vehicle leaves a cluster and enters a new one to learn information.

An attack developed by the authors of [24] aims to associate a large set of collected anonymous location samples to anonymous location profiles using the established Multiple Hypothesis Tracking (MHT) algorithm to track vehicles' locations over an

extended period of time. MHT addresses the data association problem by generating a set of data associations hypotheses every time a new set of measurements arrives, with each hypothesis being a possible association of a measurement with a target. The probability for each hypothesis to be correct is calculated and the highest probability is chosen to be the solution. MHT relies on Kalman filters [9] to estimate the state variables of each target: position and velocity. This attack assumes the role of a passive attacker with perfect eavesdropping capabilities, where an attacker receives all beacon messages sent over the network. They assume that vehicles broadcast their location and velocity at regular intervals, but with pseudonyms that change for every packet to completely anonymize the transmissions. Their experiments showed that at high beaconing rates of a beacon a second or faster and less than 100 vehicles, they were able to track vehicles for on average 800 out of the 1000 seconds in their simulations. Increasing the vehicle density to be between 100 and 250 vehicles saw the average tracking time drop to 700 seconds. A beaconing rate of a beacon every two seconds sees an average of less than 400 seconds of tracking when there are even 50 vehicles, and a drop to 150 seconds when there are 100 vehicles. Beaconing rates slower than a beacon every two seconds only saw tracking of 100 seconds when there were less than 25 vehicles in the system, and there was no substantial tracking after 50 vehicles. Their attack is also dependent on accurate position information. When they introduce random noise into the gathered position information, a random offset anywhere between one to five meters decreased tracking by 200 down to 700 seconds. They also explore the effectiveness of their tracking with varying equipment rates. They actually see an increase in average tracking time, up to nearly 900 seconds when the equipment rate is between 10 and 20%. This is due the much smaller number of vehicles being tracked, as they can only track equipped vehicles, and lower equipment rates mean the likelihood of equipped vehicles crossing paths is less likely.

While I also assume the role of a passive attacker, my attack does not assume perfect eavesdropping capabilities, only receiving packets within the range of listeners placed, nor do I assume the ability to gather location and velocity information from signal packets, simply the pseudonyms associated with the packets. In my simulation, I also use a periodic change strategy that changes pseudonyms periodically, as opposed to per packet, as pseudonym changes can be expensive, thus this is a more likely employed strategy. I also explore the effects of varying equipment rates (called penetration rates in my scenarios) in my experiments, but because I am gathering multiple protocols information (long-distance pseudonym changing and short-distance non-changing), vehicles not equipped with the short-distance non-changing equipment continue to emit long-distance pseudonym changing protocol signals introducing noise into the information gathered when trying to associate signals. The results of my experiments with varying penetration rates can be found in Chapter 5.

Chapter 3

DESIGN

In this chapter, I describe the design of the attack model I use to deanomyize vehicles and undermine their privacy by associating DSRC pseudonyms with their corresponding WiFi identifiers. This chapter outlines overall goals of the attack, requirements of the model, the traffic simulations used, and a brief design overview of the attack model. In Chapter 4, I go into more detail about the attack model implementation.

3.1 Goals

The primary goal of my attack is to correctly associate identifiers with their respective pseudonyms. This information will ultimately be used to attack users' privacy by allowing attackers to track users over an extended period of time by recreating users' paths based on the locations where signals with their identifiers or pseudonyms were seen.

3.2 Requirements

In this section, I detail the necessary system requirements for the attack model. The following are formal requirements the attack model maintains:

- The model is module and can be generalized to any long-distance pseudonym changing signal and short-distance non-changing signal configurations, and listener placement configuration.
- The model can be used on a system with varying identifier penetration rates.
- The model can quantitatively evaluate the success of its attack.

The attack model attacks vehicular privacy of nodes within a VANET system using a long-distance pseudonym changing protocol and a short-distance non-changing protocol. The model must not be constrained to specific signal types for either protocol. Thus, the range, heart rate, and change rate of the signals can be adjusted within the attack model to accommodate different signal types. The model must also be able to perform with varying listener placement configurations. I also design the attack model to perform with varying listener placement configurations to make it applicable to multiple scenarios. While the attack model benefits greatly from a greater number of listeners and a listener placement combination that maximizes the number of signal packets gathered, I show that it does not require the optimal listener placement configuration to produce substantial results. This is applicable in a real-world situation where attackers may not be able to place listeners in optimal locations nor know the optimal locations, or if attackers are interested in vehicles traveling to a specific location. Note that my attack assumes non-overlapping listener coverage. That is, listeners do not overlap in their listening range, thus a broadcast only be picked up by at most one listener. While this is not required, I chose this assumption to lower the complexity of my model. Although some areas of the attack model require knowing if vehicles were at different listeners at a given time, this information can be determined by keeping track of which listeners overlap in coverage. Furthermore, overlapping listener coverage could potentially provide more useful information, such as a basic form of triangulation by using the overlapping coverage area to further narrow down the exact location of a vehicle as opposed to using an entire listener's range, or using the overlapping coverage area as another region in conjunction with the corresponding listeners' non-overlapping areas.

Another requirement is that the attack model must still be relatively successful regardless of radio market penetration rate, that is, the percentage of vehicles within the VANET system that support a given signal type. This is important because

there are some real-world scenarios where not all vehicles in the area of observation will support all signal types. Therefore, not requiring all vehicles within the VANET system to have both signal types makes my attack more broadly applicable to a realistic scenario.

The final requirement is that I must be able to quantitatively measure the success of my attack to allow for input parameter tuning and to conclude whether my attack was successful. This also allows us to compare my attack results against other known attacks [6, 3, 5, 24].

3.3 Vehicle Simulation Model

For my experiments, I use a DSRC signal for the long-distance pseudonym changing signal and a WiFi protocol signal for the short-distance non-changing signal. The DSRC protocol signal has a range of 100 meters, a heart rate of 0.1 seconds, and change frequency of 30 seconds. The WiFi protocol has a range of 25 meters and a heart rate of 30 seconds. While I use DSRC and WiFi, the range, heart rates, and change frequency of either protocol can be configured to emulate other signal types.

For listener placement, I use a handpicked approach where I manually select and place listeners at locations with a large amount of vehicle traffic to increase the number of radio packets gathered [5]. I also ensure that listeners do not overlap in coverage, as my attack methods work under the assumption that listeners' coverages are disjoint. While listener placement can be optimized using techniques such as a genetic algorithm to find the optimal listener placement combination that maximizes the number of packets gathered, and there is a separate Cal Poly student project that explores this option, maximum packet gathering is not the goal of this attack. Therefore, a handpicked listener placement technique is adequate for my attack. This also provides a realistic scenario, where an attacker can simply set up listeners at lo-

cations where they know to have the most traffic or locations that they are specifically interested in.

I use SUMO [2] to simulate realistic traffic on a number of real-world maps, as opposed to research that uses grids in their experiments [6]. Using the DSRC and WiFi protocols as my two signal types, I then run simulations using the vehicles paths generated by SUMO with varying degrees of WiFi penetration rates, that is a percentage of my vehicles will support both WiFi and DSRC protocols, while the rest will only support DSRC; my simulations maintain a 100% DSRC penetration rate. I choose a 100% DSRC penetration rate because of its many VSC usages. Therefore, it is likely to be federally mandated in the future. Furthermore, while my simulation maintains a 100% DSRC penetration rate, it is not required, as WiFi is the unique identifier of vehicles and DSRC pseudonyms simply help to expand our area of knowledge from created associations. Results of my experiments running the attack without any DSRC-to-DSRC associations illustrate this idea in Chapter 5. Varying WiFi penetration rates serve to illustrate the effectiveness of the attack, regardless of penetration rate and is representative of a real-world scenario, as some vehicles may not support all protocols. Lower WiFi penetration rates are theoretically more difficult to attack, as I leverage the WiFi identifiers to uniquely identify vehicles and determine the number of vehicles. DSRC packets received from vehicles without WiFi capability are essentially noise in the system, as I have no way of identifying the vehicle they originated from, but the attack model still tries to associate them to WiFi identifiers, as I do not know if a DSRC packet originated from a vehicle with or without WiFi capability. Using the SUMO generated traffic simulations, I step through each vehicle's path, stopping at intervals in accordance to the two signal protocol heart rates and simulate a signal broadcast. A broadcast is simulated by checking if any listener is within the broadcast range of a vehicle when it broadcasts

a signal and recording that the listener, if within range, received a packet tagged with the respective signal's pseudonym or identifier at the given time.

3.4 Brief Overview of Attack Model

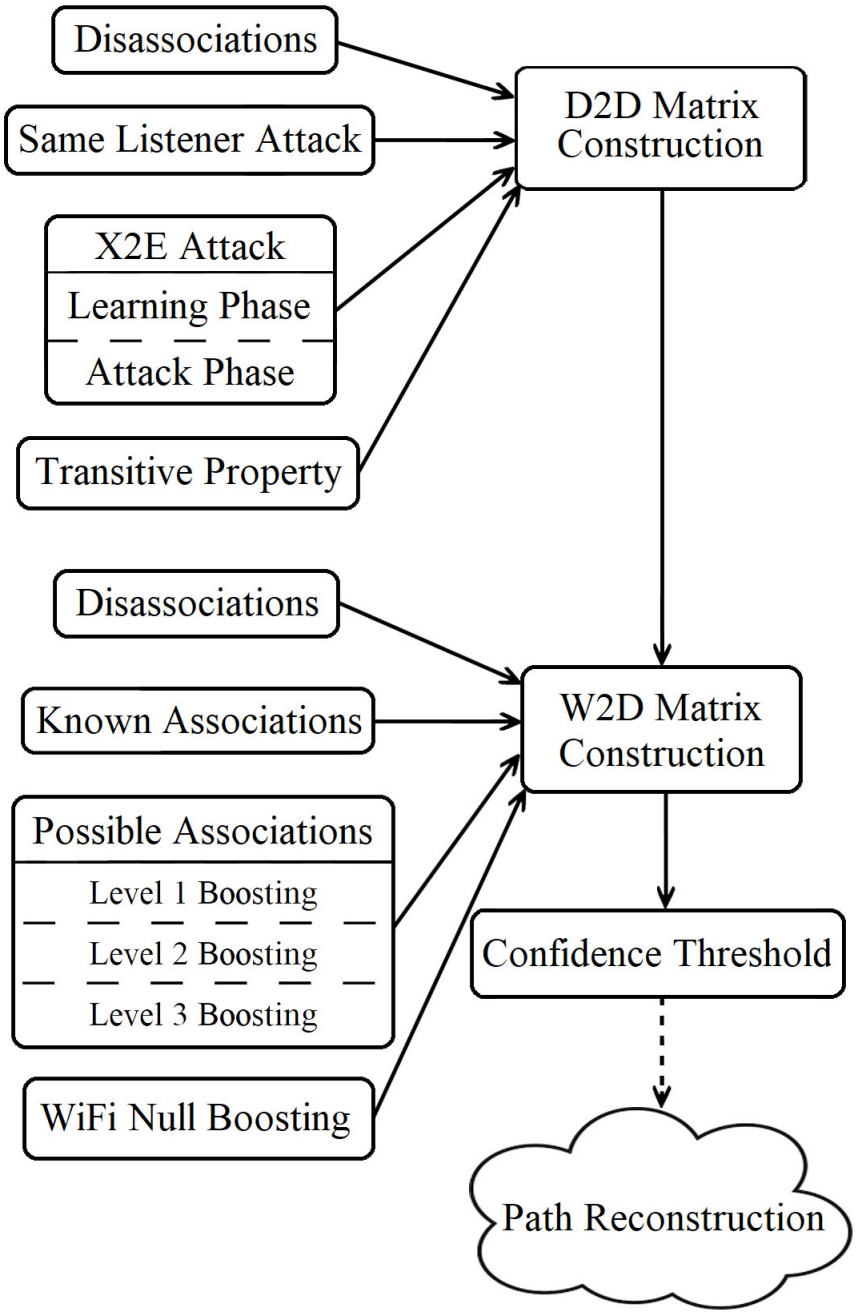


Figure 3.1: Attack Model

The attack is divided into the construction of two association matrices: DSRC-to-DSRC (D2D) and WiFi-to-DSRC (W2D). The D2D matrix is an MxM matrix, where M is the number of DSRC pseudonyms observed by listeners and is used in a transitive property manner when creating WiFi-to-DSRC associations and disassociations in the W2D matrix construction. The W2D matrix is an $(N + 1)$xM matrix, where N is the number of WiFi identifiers observed by listeners. I add an extra row to the W2D matrix that represents all vehicles without WiFi capability, dubbed WiFi Null. Figures 1.1 and 1.2 show the initialization of the D2D matrix and W2D matrix, respectively. When making WiFi-to-DSRC associations, note that WiFi identifiers have a one-to-many relationship with DSRC pseudonyms, where a WiFi identifier can be associated with many DSRC pseudonyms, but a DSRC pseudonym can only be associated with a single WiFi identifier. DSRC pseudonyms have a many-to-many relationship with other DSRC pseudony, where a DSRC pseudonym can be associated with many other DSRC pseudonyms. Furthermore, if the WiFi penetration rate is lower than 100%, then some DSRC pseudonyms cannot be correctly associated to any WiFi identifier, as some pseudonyms will originate from a vehicle without WiFi capability. In the remainder of this section, I explain the different steps involved in the construction of each matrix. I design the attack model to be modular such that steps can be added to or removed from either matrix constructions if new research conceives new attacks that can be incorporated into the attack model. Results of my experiments using different configurations of the D2D matrix construction illustrates this idea and can be found in Chapter 5.

3.4.1 D2D

Within the D2D matrix, for any pair of DSRC pseudonyms my attack will conclude that the two pseudonyms are: disassociated, associated, possibly associated, or that there was not enough information to make a conclusion. The D2D matrix construction

is broken into four steps: finding disassociations, the Same Listener Attack, the Exit-to-Enter (X2E) Attack, and applying the transitive property to disassociations and associations. These steps are illustrated in Figure 3.1.

The first method that I use labels a pair of DSRC pseudonyms to be disassociated. A pair of pseudonyms are labeled as disassociated when the pseudonyms are seen at different listeners in an overlapping time interval. Because pseudonyms are assumed to be unique across all vehicles, a vehicle can only use one pseudonym at a time, and listeners do not have overlapping coverage, two pseudonyms can be labeled as disassociated if they are observed at different locations at the same point in time. An example of this is illustrated in Figure 1.3, where d_1 and d_2 are seen at different locations at the same time, therefore d_1 and d_2 are disassociated.

The next method, dubbed the *Same Listener Attack*, finds DSRC-to-DSRC associations. In the Same Listener Attack, I look for a pseudonym that is first seen immediately after another pseudonym is last seen. If both observations occur at the same listener, then there is a high probability these observations can be explained as the originating vehicle changing pseudonyms while still in the range of the listener. Therefore, I am able to, with high precision, associate the two observed pseudonyms to each other. This type of attack was previously described in [5].

The next method, dubbed the *Exit-to-Enter* (*X2E*) *Attack*, finds DSRC-to-DSRC associations and possible associations. This attack is an extension of the method used in [6, 3]. Enter and exit events are defined as when a vehicle enters or exits my area of observation, respectively. Examples of enter and exit events are illustrated in Figure 3.2. This method is broken up into two phases: a learning phase and an attack phase. During the learning phase, I observe the number of vehicles that go between any two pairs of listeners, and the average time each vehicle took to make that trip. The attack phase then tries to match enter events to their respective

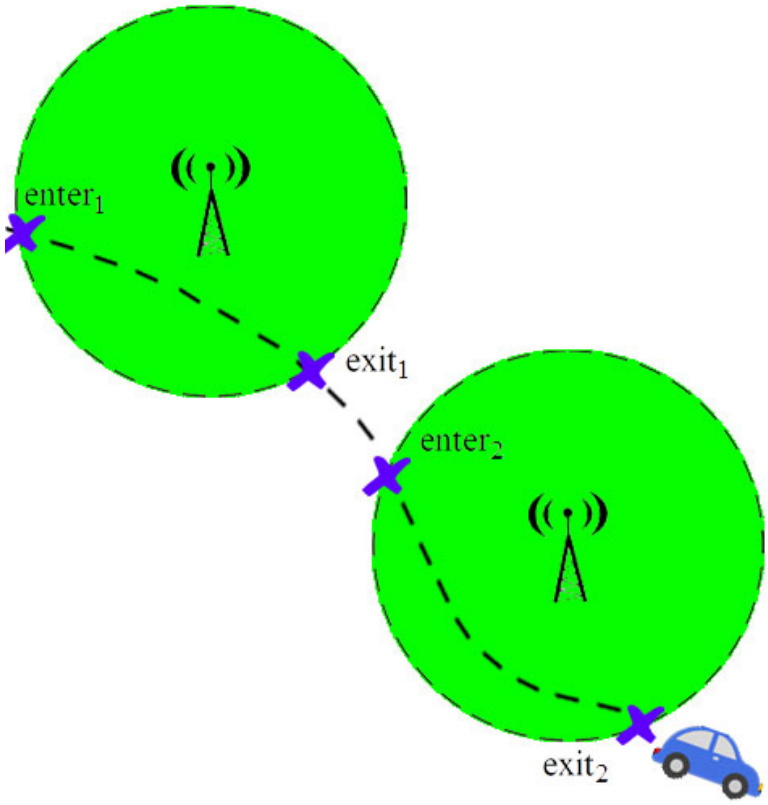


Figure 3.2: Enter and Exit Events of Example Vehicle Path

preceding exit events by constructing a bipartite graph with exit events in one set and enter events in the other. Edges are assigned from an exit event to an enter events with a weight determined by the previously learned information. Note that I use negatively weighted edges, that is, the edges that are more likely to be correct connections between an exit and an enter event have lower valued weights than edges corresponding to less likely correct connections. An example bipartite graph is shown in Figure 3.3. I then solve the minimum weight matching or linear assignment problem on the graph. The linear sum problem is a combinatorial optimization problem that finds the minimum weighted perfect matching of a weighted bipartite graph. That is, the goal is to find the minimum sum of weights of edges while having each item on one side of the bipartite graph assigned to only a single item on the other side of the graph. I use negative weights due to Python's NumPy package having a function that

solves minimum weight matching, but not maximum weight matching. Then I take each in the solved graph, and if its weight is a percentage of the maximum weight recorded within the graph, then I set the edge's corresponding enter and exit events' pseudonyms to be associated. If the edge's weight does not meet this criterion, then I set the two pseudonyms as a possible association. The effectiveness of this method is affected by listener placements. I saw that when listeners are placed near each other in high traffic locations and not spread throughout the area of attack, this method yields little to no data, most likely due to all edges being of equal weights (i.e., vehicles are equally likely to travel from any one listener to any other listener). Therefore, solving the minimum weight sum of the bipartite graph has extremely low precision. The relationship between this attack method and listener placements is not the focus of this project, therefore I do not look into what sorts of listener placement configurations maximize the efficacy of this attack method.

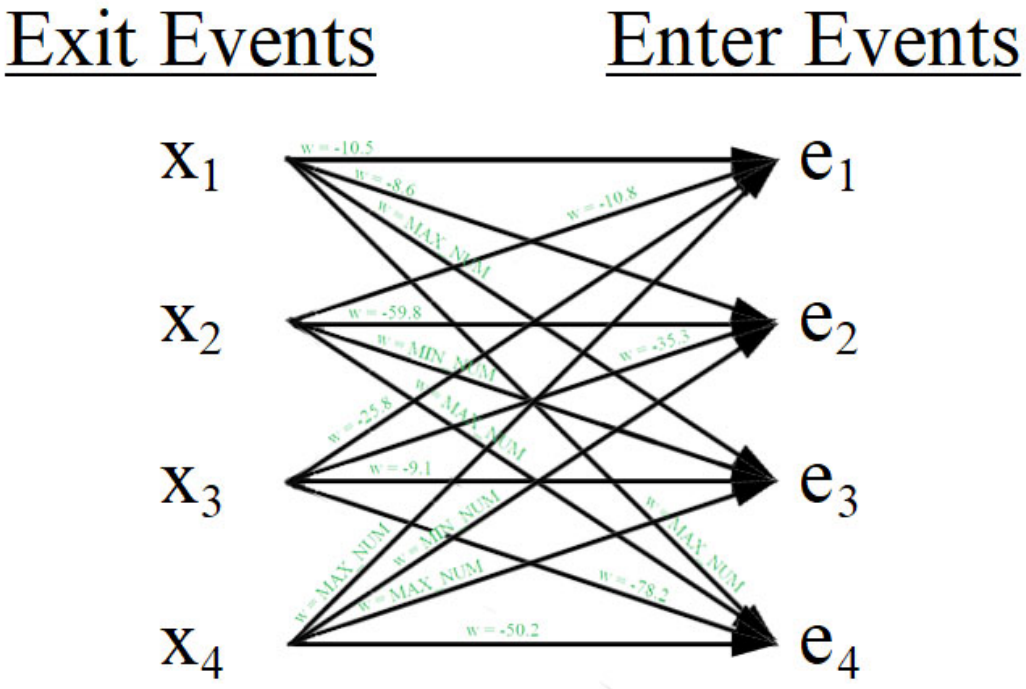


Figure 3.3: Weight Bipartite Graph constructed in X2E Attack

Finally, the transitive property is applied across the D2D matrix. That is, if d_1 is associated to d_2, and d_2 is associated to d_3, then all three pseudonyms are said to be associated to each other. I apply the transitive property only on disassociations and associations and not possible associations because through my experiments the high uncertainty of possible associations caused a cascading number of incorrect associations when a single incorrect association bridged two association groups.

The D2D matrix is then used in the W2D matrix construction. It is used in a transitive property manner in creating WiFi-to-DSRC disassociations and associations. For example, if DSRC pseudonyms d_1 and d_2 are associated, and WiFi identifier w_1 is associated to d_1, then w_1 is also associated to d_2.

3.4.2 W2D

Within the W2D matrix, the element at row i and column j (denoted as $element_{ij}$) contains the probability that WiFi identifier w_i is associated to DSRC pseudonym d_j. Note that, since DSRC pseudonyms can only be associated to a single WiFi identifier, each column within the matrix sums to one and are mathematically independent of each other. Also, since a WiFi identifier can be associated with many DSRC pseudonyms, rows can sum to any arbitrary value. When w_i and d_j are said to be seen together, or that w_i is seen with d_j, it means that they were seen by the same listener with an overlapping time interval.

When updating W2D probabilities, I employ a method I have dubbed boosting multiple times. Boosting is a method done in terms of a given DSRC pseudonym, where I take a portion of some elements in the pseudonym's corresponding column and redistribute them among elements in the same column due to having information that implies that the boosted elements should have a higher value, that is the WiFi-DSRC pairs that the boosted elements represent are more likely to be associated than

the pairs represented by elements not being boosted. Boosting is done within each DSRC column independently. Thus, a single boosting event for a pseudonym only affects elements in the corresponding matrix column. Therefore, I say that for a given DSRC pseudonym d_j, I boost all WiFi signifiers w_i in set W, where W is the set of WiFi signifiers that I have concluded to be likely associated to d_j, and take the boost from WiFi signifiers w_k in set W', where W is the set of WiFi signifiers that are less likely to be associated to d_j. Note that W' is not necessarily simply $W_{all} - W$ where W_{all} is the set of all WiFi signifiers that I have observed. I define the total sum of the values for W' to be Σ, and Δ to be the redistribution factor, that is the percentage taken from Σ to be redistributed to the boosted values. When boosting, for each column, each element corresponding to identifiers in W' will be reduced by (1- Δ) (multiplication) and each element corresponding to identifiers in W will be increased by $\Delta * \Sigma$ (addition). An example illustration of boosting can be found in Figure 4.2 where for d_1 I boost w_1, w_2, and w_3, while taking the boost from w_4 and w_5. Tuning the redistribution factor using methods, such as machine learning, can increase the effectiveness of the attack model, but it is outside of the scope this project.

Each element is initialized to be equal to $\frac{1}{N+1}$, where N is the number of WiFi identifiers that I have observed, that is, each DSRC pseudonym is initially equally likely to be associated to any WiFi identifier. I then find WiFi-DSRC disassociations. The same criteria for disassociation from the D2D matrix creation are used here. A WiFi identifier and a DSRC pseudonym are found to be disassociated if they are seen at different listeners while having an overlapping time interval. When a WiFi identifier w_i is found to be disassociated from DSRC d_j, then $element_{ij}$'s value is set to zero, and its original value is distributed among all other elements within column j whose current value does not equal zero. This redistribution represents when a DSRC pseudonym is found to not possibly be associated to a given WiFi identifier. In this

case, the likelihood that the pseudonym is associated to any other WiFi identifier increases.

Next, I find known associations between a WiFi identifier w_i and a DSRC pseudonym d_j. The criteria for a known association are that w_i and d_j are seen together and there are no other pseudonyms seen during that time. While theoretically this method may associate some pairs that are not actually associated, my experiments have found that it is a simple approach that yields a fairly large number of true-positive associations with high precision. When a known association is found between w_i and d_j, $element_{ij}$ is set to one and all other elements in column j are set to zero. Each $element_{\text{ij}}$ is set to 0, where $d_{j'}$ is each pseudonym disassociated to d_j. Their original values are redistributed among all other elements in their respective columns that do not currently equal zero.

I then find possible associations between a WiFi identifier w_i and a DSRC pseudonym d_j. Possible associations are incorporated into the W2D matrix through the use of boosting. There are three levels of possible associations, each with their own boosting phase, where subsequent levels are less likely, and therefore will receive a smaller boost than preceding layers. In my current attack model, each level uses a different redistribution factor Δ. As noted before, these redistribution factors can be tuned to maximize effectiveness, but generally I saw that using a larger Δ in the second level and a smaller Δ in the last level yielded more favorable results than using the same Δ across all levels.

The first set of possible associations is associations between w_i and d_j where w_i was seen with d_j, but there were other pseudonyms also seen at the same time. Therefore, if W is the set of all WiFi signifiers seen with a DSRC pseudonym d_j, then d_j is equally likely to be associated to any signifiers in W, given that d_j is only seen with each signifier in W once. I also consider the number of times d_j is seen with w_i.

That is, if d_j is seen with w_i twice and w_k once, then $element_{ij}$ should get twice the amount of boost as $element_{kj}$. For a given pseudonym d_j, I boost all identifiers in W, where W is the set of all WiFi identifiers seen with d_j and take the boost from W', where W' is the set of all WiFi signifiers not seen with d_j.

The second set of possible associations uses DSRC-to-DSRC associations found in the D2D matrix construction. These associations are based on signals seen together by a single degree of separation. That is, w_i and d_j are possibly associated if w_i was seen with d_k, but other pseudonyms were also seen at the same time, where d_k is a DSRC pseudonym associated to d_j. Like the previous level of boosting, I also consider the number of times signals were seen together, giving those that were seen together multiple times a larger boost than those that were seen fewer times together. For a given pseudonym d_j, I boost all identifiers in W, where W is the set of all WiFi signifiers seen with a DSRC pseudonym d_k, and take the boost from W', where W' is the set of all WiFi signifiers not seen with d_j or not seen with d_k, where d_k is a DSRC pseudonym associated to d_j. At this level, I use a different Δ than the previous level, called Δ', which is the minimum between 1.5*Δ and 0.99. This larger Δ value is due to boosting's effectiveness being based on Σ, which is the sum of all elements corresponding to elements less likely to be associated. Because this is the second level of boosting, the less likely associated corresponding elements have already been decreased in the first level of boosting, resulting in a smaller Σ. So, for the boosting to still be effective, I need to increase Δ to increase the amount of value that I "take" from Σ to be redistributed. Note that at this step, when evaluating the relationship between w_i and d_j, I also look to see if d_j is associated with any other DSRC pseudonym d_k that is known to be associated to w_i (i.e. w_i and d_k were seen together with no other pseudonyms around). If this is the case, I automatically set w_i and d_j to be a known association, ignoring any other identifiers that d_j was seen with and not boosting any identifiers at this level with regards to pseudonym d_j. I also

disassociate w_i from any pseudonym that is disassociated from d_j and redistribute the respective probability value among all elements in the corresponding column that do not currently equal zero.

The final set of possible associations uses DSRC-to-DSRC possible associations found in the D2D matrix construction. Using the same degree of separation logic as before but using DSRC pseudonyms that are possibly associated, I consider a pair w_i and d_j to be possibly associated if w_i was seen with d_k, where d_k is a DSRC pseudonym possibly associated to d_j. At the previous level, for a given pseudonym d_j, boosting any WiFi identifiers possibly associated to d_j is superseded by an automatic known association between w_i and d_j if there existed a d_k that was the only seen DSRC pseudonym with w_i at a single listener (i.e. w_i is associated to d_k and d_k is associated to d_j, therefore w_i is also associated to d_j). However, due to the low precision of possible associations, applying the same transitive automatic associations logic as the previous level would yield too many false positives. Therefore, because I still want to give some credence to these conjectures, I treat a WiFi identifier that would have been automatically associated as simply another identifier that was seen with a pseudonym that is possibly associated to the given pseudonym, adding it to the set of identifiers to be boosted. In addition to this, I also use a different Δ" than the previous levels. Δ'' is equal to 0.75*Δ due to the low precision and thus confidence of the D2D possible associations used to extend boosting at this level. Similar to the previous levels of boosting, I again take into account the number of times a pair of signals were seen together, giving a larger boost to those seen more times together than those seen fewer times together. For a given pseudonym d_j, I boost all identifiers in W, where W is the set of all WiFi identifiers seen with d_k, and take the boost from W', where W' is the set of all WiFi signifiers not seen with d_j, or d_k or d_l, where d_k is a pseudonym associated to d_j and d_l is a pseudonym possibly associated to d_j.

When the WiFi-to-DSRC association matrix is fully constructed, I look for any probabilities that meet a certain confidence threshold. Using a minimum confidence threshold of 50%, because each column sums to one, a probability of at least 50% means that the element's corresponding WiFi-DSRC association contains the majority. The results for different levels of confidence thresholds is presented in Chapter 5.

3.4.3 WiFi Null Boosting

A problem arises when an attacker does not know how many vehicles in the area of observation have WiFi capability, that is an attacker is unable to determine the number of vehicles with WiFi capability and the number of vehicles without. One approach that I have explored is a method I dubbed WiFi Null Boosting. WiFi Null Boosting follows the same boosting principle as aforementioned, where for a given DSRC pseudonym d_j, I increase the value of all elements $element_{ij}$ where WiFi identifier w_i is an identifier more likely to be associated to d_j than an identifier that is not being boosted. In the case of WiFi Null Boosting, the WiFi identifier used is WiFi Null. Therefore, each time a DSRC pseudonym d_j is seen at a listener for an entire period without any overlapping WiFi signals around, then the probability that d_j is associated to WiFi Null (i.e. d_j originated from a vehicle without WiFi capability) is boosted, where the boost is taken from all other probabilities in the given column. WiFi Null Boosting uses its own redistribution factor (WiFi Null Δ) that is different from the main delta value. Unfortunately, my experiments regarding this method yielded little results. Due to the DSRC protocol's high frequency heart rate and WiFi protocol's low frequency heart rate, many more pseudonyms were seen without an overlapping WiFi than the number of pseudonyms actually originating from a vehicle without WiFi. This, in combination with the relatively low change-rate of the DSRC pseudonym in my simulation, made it extremely difficult to determine the correct pseudonyms. Thus, boosting with a small redistribution factor yielded no

positives (true or false), and boosting with a medium or large redistribution factor yielded too many false positives, and ultimately detracted from the number of correct WiFi-to-DSRC associations.

Chapter 4

IMPLEMENTATION

In this chapter, I expand on the design overview detailed in Chapter 3 by providing pseudocode for the algorithms used in the attack and explaining in more detail the reasoning behind many of the implementation choices.

4.1 Terminology

I begin with definitions of commonly used terms in my attack. Listeners are the devices used to pick up radio signals. I assume perfect packet reception, therefore if a listener is within range of a vehicle when it emits a signal, then the listener picks up the packet. Listeners record packets and their pseudonym or identifier in terms of observation intervals. Observation intervals contain the listener it was recorded at, the pseudonym or identifier the packet used, the protocol used, the time a pseudonym was first seen at a given listener, and the last time it was seen. The observation interval's start and stop times are recored on the basis that the listener receives a continuous stream of packets with a pseudonym or identifier over a given time. Since packets are discrete, I allow for a gap between packets. This gap can be adjusted depending on the protocols that are being used. Therefore, if a vehicle leaves a listener's range, then returns some times later still using the same pseudonym, the listener will create two separate observation intervals for that vehicle.

We define overlapping as the following when used to describe listener coverage, time intervals, and observation intervals. Overlapping listener coverage and overlapping time interval definitions are straightforward. Overlapping listener coverage is when an area is in the range of multiple listeners, thus those listeners overlap their

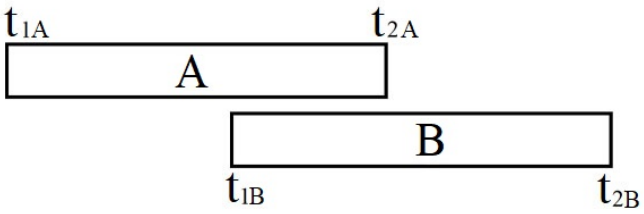


Figure 4.1: Example of Observation Intervals Having Overlapping Time Intervals

coverage. Overlapping time intervals is when the time intervals of multiple observation intervals overlap in time. An example of overlapping time intervals is illustrated in Figure 4.1, where observation intervals A and B have overlapping time intervals because B's start time t_{1B} is less than A's end time t_{2A}, meaning B began before A ended. Overlapping observation inteverals is when observation intervals both overlap in time and are recorded at the same listener. Therefore, for a given observation interval, any other observation interval is said to be overlapping if they have overlapping time intervals that occurred at the same listener.

The observation area is the combined coverage area of all listeners. For my work, I consider mix zones as areas outside of the area of observation, where vehicles can mix and change pseudonyms without the attacker's knowledge.

Enter and exit events are used in the Exit-to-Enter Attack as part of the D2D matrix construction. These terms were previously defined in [6] and [3] as when vehicles enters or exit a mix zone, respectively. However, in this paper I define enter and exit events in terms of the area of observations, that is, enter and exit events are when a vehicle enters or exits the area of observation, respectively. Therefore, an enter event is when a vehicle exits the mix zone and enters the area of observation. An exit event is the opposite, when a vehicle enters the mix zone and exits the area of observation. An illustration of enter and exit events can be found in Figure 3.2. As per the illustration, the moment a vehicle comes within range of a listener is the enter event, and when it leaves the range of a listener is the exit event. Therefore, an enter

event can be recorded as the start of an observation interval, and an exit event as the end of an observation interval. Note that in the illustration I mark the locations of the enter and exit events. Due to my assumption that attackers are not able to read the location data associated with a pseudonym, I cannot actually pinpoint the exact locations of exit and enter events, simply record the listener's location at which the events occurred.

I define association as originating vehicle. Therefore, a pseudonym is associated to another pseudonym or an identifier if the two signals originate from the same vehicle.

I refer to my two matrices as association matrices. An association matrix is a matrix where $element_{ij}$ is a value representing the association status between its corresponding row and column values. The D2D association matrix contains values that represent the type of association between two pseudonyms (e.g. -1 = no information, 1 = association). The W2D association matrix contains values that represent the probability of an identifier and pseudonym being associated.

I define boosting as the act of taking a percentage of elements in the W2D association matrix and redistributing the taken amount to other elements. Boosting is done in terms of DSRC pseudonyms in the W2D matrix construction, that is boosting happens independently in each column. When I conclude that WiFi identifier in set W are more likely to be associated to a given DSRC pseudonym d_j than WiFi identifiers in a separate set of WiFi identifiers W', then I say that for d_j, I am boosting the identifiers in W and taking the boost from identifiers in W'. I boost the identifiers in W by increasing the value of elements in column j that correspond to those identifiers, and I take the boost from identifiers in W' by decreasing the same amount as the total amount given to the boosted identifiers. The redistribution of values by increasing the probability that pseudonym d_j is associated to an identifier in W and decreasing the probability that it is associated to an identifier in set W' is to reflect

in the matrix the conclusion that some identifiers are more likely to be associated to d_j and others are less likely. Note that the set *W'* is not necessarily simply $W_{all} - W$, where W_{all} is the set of all WiFi identifiers seen across all listeners. An example of boosting is shown in Figure 4.2. In this example, d_1 is more likely to be associated to w_1, w_2, and w_3 and less likely to be associated to w_4, and w_5. Therefore, I will boost the former set and take the boost from latter set. Σ is defined to be the summation of elements corresponding to identifiers in *W'*. Δ is the redistribution factor, that is the percentage taken from Σ to be redistributed to the boosted values. The redistribution is not necessarily equal. As seen in Figure 4.2, the portion given to each boosted value is multiplied by a respective c value and divided by the summation of all c values. The variable c allows us to boost some identifiers in W more than others, based on how often each identifier meets the boosting criteria. For example, if I am boosting identifiers that are seen with pseudonym d_j, then an identifier's c value is the number of times the identifier was seen with d_j. Thereby, identifiers seen more times with d_j will have a larger c value, and get a larger boost than those seen less often and with a smaller c value.

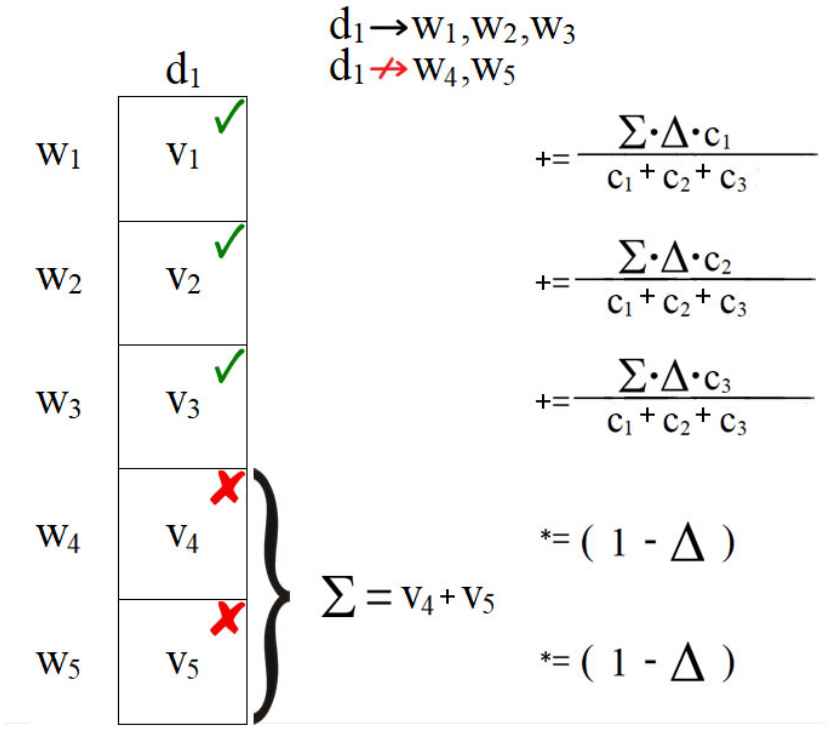


Figure 4.2: Example of Boosting

4.2 Attack Model Breakdown

I design the attack model by constructing two association matrices: DSRC-to-DSRC (D2D) and WiFi-to-DSRC (W2D). I process the information that listeners have gathered to create two hash maps *dgroups* and *wgroups*. I use *dgroups* to map each DSRC pseudonym to a list of its corresponding observation intervals, and *wgroups* to map each WiFi identifier to a list of its corresponding obersvation intervals. These maps are used when constructing the two matrices.

The following sections detail the construction process of both the D2D association matrix and the W2D association matrix.

4.3 DSRC-to-DSRC

The D2D matrix is an MxM matrix where M is the number of unique DSRC pseudonyms observed across all listeners. Within the matrix, $element_{ij}$ corresponds to the association status between pseudonyms d_i and d_j. In my model, this status is represented by four possible values: -1 (no information), 0 (disassociated), 1 (associated), or 0.5 (possibly associated). Note that 0.5 is simply a constant value that we have chosen to signify possible associaiton. Future techniques may distinguish different possible associations by their confidence, and have possible associations be represented by a number between 0 to 1 (exclusive). But as I currently have only a single form of making possible DSRC-to-DSRC association, I simply chose 0.5, which is halfway between 0 and 1. At the start, all elements are initialized to -1, except for the diagonal which is initialized to 1 (i.e., a pseudonym is associated to itself). After the initialization, the matrix construction is broken into a series of steps; theses steps are finding disassociations, the Same Listener Attack, the X2E Attack, and applying the transitive property across the matrix. I explain each of these steps in more detail in the following subsections, with pseudocode for the entire D2D matrix construction algorithm in Algorithm 5. After construction, it is used in a transitive manner when making W2D disassociations and boosting. For example, if DSRC pseudonyms d_1 and d_2 are disassociated and w_1 is found to be associated to d_1, then w_1 cannot be associated to d_2 (disassociated). I present the results of my experiments using different components of the D2D matrix construction in Chapter 5.

4.3.1 Disassociations

I begin by finding disassociations. I conclude that two pseudonyms d_i and d_j are disassociated if they have overlapping time intervals at different listeners. This implies that they were at two different locations during the overlapping time because I

assume non-overlapping listener coverage. If they are at the same listener, then they are either considered associated or possibly associated, which I explain in the following subsections. To find disassociations, I iterate through the set of pseudonyms, comparing each pseudonym d_i's observation intervals with every other pseudonym d_j's oberservation intervals, looking for the criteria for disassociation. Note that we assume unique pseudonyms across all vehicles.

Input: dgroups: Dictionary mapping DSRC pseudonym to list of observation intervals

Result: Updates D2D matrix with associations found

dobis ← set of all observation intervals

dobis.sort() group by pseudonyms, then order by ascending time order

foreach *DSRC psuedonym* $d_i \in dobis$ **do**

  start ← first time and location d_i is seen

  end ← last time and location d_i is seen

  record *start* and *end* for d_i

end

foreach *DSRC psuedonym* $d_i \in dobis$ **do**

  foreach *DSRC pseudonym* $d_j \in dobis$ *that is not equal to* d_i **do**

    if $d_i.location = d_j.location$ *and* $d_j.start$ *is* ***DSRC-HEARTRATE*** *seconds after* $d_i.end$ **then**

      $mtx[i, j] \leftarrow mtx[j, i] \leftarrow 1$

  end

end

Algorithm 1: SAME_LISTENER_ATTACK finds associated DSRC pairs based on a DSRC appearing immediately after another DSRC is last seen

4.3.2 Same Listener Attack

In the *Same Listener Attack*, I create associations between pseudonyms using the algorithm found in Algorithm 1. The Same Listener Attack leverages knowledge of the protocol heart rates and is based on the idea that if a vehicle were within range of a listener, then when the vehicle uses the new pseudonym for the first time, to the listener it will appear as if a new vehicle entered its are of observation. Therefore, I look for pseudonyms that first appear at listeners where pseudonyms were last seen. I start by taking the set of all observation intervals, grouping them by pseudonyms, then sorting them by ascending time order. I then record when and where the first and last time each pseudonym is seen. I then iterate through all pseudonyms, checking to see if any of them are first seen *DSRC heart rate* seconds after any other pseudonym is last seen. Any pair of pseudonyms that meet these criteria are labeled as associated in the D2D matrix.

4.3.3 X2E Attack

The next attack is dubbed the *X2E* (*Exit-to-Enter*) *Attack*. This algorithm is based on the attacks used in [6] and [3], where an attack tries to match exit events to their corresponding enter events. This is done by creating a weighted bipartite graph for the set of exit events and the set of enter events and solving the linear sum problem to get a minimal sum cost perfect match of the graph. The attack is broken up into two phases: a learning phase and an attack phase. I add improvements to both phases, which I explain in each of their respective subsections.

The following subsections detail the learning phase and attack phase of the X2E Attack.

Input: dgroups: Dictionary mapping DSRC pseudonym to list of observation intervals

Output: Mapping of pair of listeners (A,B) to number of vehicles traveling from A to B and combined time it took

Extract *subject_id*'s from observations intervals in dgroups and consolidate by *subject_id sgroups* ← mapping subject_id to list of respective observation intervals *data* ← empty hash map

foreach *subject_id* s_i, *list of observation intervals obis* ∈ *sgroups* **do**

    obis.sort() sort by ascending time order

    foreach *pair of observation intervals* $(o_1, o_2) \in obis$ **do**

        if $o_1.start < o_2.start$ **then**

            $key \leftarrow (o_1.loc, o_2.loc)$

            data[key] add 1 to count of number of vehicles, add (o_2.start - o_1.end) to total time

    end

end

return data

Algorithm 2: LEARNING_PHASE Gathers data needed to create weighted edges in Attack_Phase of X2E_Attack

Learning Phase

During the learning phase, I record the number of vehicles that traveled between two listener locations and their total travel time. I use this information to assign weights to the edges between exit and enter events. I illustrate my algorithm in Algorithm 4. Previous implementations of the learning phase realisticly limit their knowledge by treating each pseudonym as a separate vehicle. This leads to data skewed towards a greater number of recorded vehicles than there actually are [6]. For

my implementation. Because it is a simulation, I do a lookup of the pseudonyms to their corresponding vehicles, eliminating the problem of double-counting pseudonyms from the same vehicle towards the total vehicle count. I chose this to illustrate the best case scenario, and I argue that an attacker can simulate traffic on their target area to get data comparable to ours. Previous attacks have also used this in an online (real-time) attack, where data collected during the learning phase is not attacked, but simply used to created the bipartite graph. Because my attack model is targeted towards a passive attacker, I use my entire attacked simulation for the learning phase. I then try to attack the same simulation during the attack phase. I chose this, again, to illustrate the best case scenario for an attacker.

The learning algorithm begins by consolidating all observation intervals belonging to pseudonyms from the same originating vehicle. For each subject vehicle, I sort its observation intervals by ascending time order to recreate a vehicle's path. I create a hash map mapping a pair of listeners (A, B) to a tuple containing the number of vehicles that went from A to B and the total time all vehicles took. I do this by going through every permutation of size two (o_1, o_2) of every vehicle's list of observation intervals. If o_1 occurs before o_2, then I use their locations as the key into the map, increase the count of number of vehicles by one, and add the difference between o_2's start time and o_1's end time (i.e., the travel time of the vehicle leaving o_1's location and arriving at o_2's location). I then return the hash map to calculate weights between exit and enter events during the attack phase.

Weights

Weights are assigned to an edge between each exit event and each enter event within the bipartite graph of the X2E Attack. The weight of an edge connecting an exit event x and an enter event e represents the likelihood that the vehicle leaving x re-

entered the observation area at e. In the example illustration in Figure 3.2, $enter_2$ is the corresponding enter event to $exit_1$. My goal is to give the edge connecting $exit_1$ to $enter_2$ an appropriate weight such that it will be chosen when solving the linear sum problem on the bipartite graph. Note that I use negative weights to signify greater likelihood due to Python's NumPy package having a builtin linear sum problem solution, and not a maximal sum cost solution. The weight algorithm I use is based on the research of [6] and [3], but I improve it by using previously found associations and taking into account the actual travel time between events to get more accurate results. My improvements increased the association recall, while maintaining the same precision.

The weight of an edge between exit event x and enter event e are determined by a set of criteria, for which the pseudocode can be found in Algorithm 3. If e occurred before x, then I choose a weight such that the edge will not be chosen (i.e., MAX_NUM, where MAX_NUM is an extremely large number). If x and e have the same pseudonyms, then choose a weight such that the edge will be chosen (i.e., MIN_NUM). If the pseudonyms corresponding to x and e have already been found to be associated, then I choose a weight such that the edge will be chosen (i.e., MIN_NUM, where MIN_NUM is an extremely large negative value); and vice versa, where if the pseudonyms have already been found to be disassociated, then I choose a weight such that the edge will not be chosen (i.e., MAX_NUM). If during the learning phase I saw no vehicles travel from x's location to e's location, then I give it a relatively small weight such that the edge may still be chosen, but unlikely (i.e., -0.1). If none of the previous conditions were met, then I give the edge a weight using the learned data. Given that n_0 is the number of vehicles that went from x's location to e's location, and t_0 is the average time they took ($\frac{total\ time\ vehicles\ took\ to\ travel}{n_0}$), and t_{delta} is the actual travel time (e's timestamp substract x's timestamp), I first calculate t_{error} as either $2-\frac{t\_delta}{t_0}$ if t_{delta} is less than t_0, or as $\frac{t\_delta}{t_0}$ if t is greater

Input: x: exit event, e: enter event, learn: map representing learned data from Learning_Phase(), mtx: current D2D matrix

Output: weight inversely relational to likelihood exit and enter events are associated (smaller number → more likely)

if *e occurredbefore x then* **then**

  $w \leftarrow MAX\_NUM$

else if *x and e have the same pseudonym* **then**

  $w \leftarrow MIN\_NUM$

else if *x.pseudonym and e.pseudonym have already been found to be disassociated* **then**

  $w \leftarrow MAX\_NUM$

else if *x.pseudonym and e.pseudonym have already been found to be associated* **then**

  $w \leftarrow MIN\_NUM$

else if *learn saw no vehicles going from x.location to e.location* **then**

  $w \leftarrow -0.1$

else

  $n_0 \leftarrow$ number of vehicles that went from *x.location* to *e.location*

  $t_0 \leftarrow$ (total time vehicles took to travel) / n_0

  $t_{delta} \leftarrow e.time - x.time$

  if $t_{delta} < t_0$ **then**

    $t_{error} \leftarrow 2$ - t_{delta} / t_0

  else

    $t_{error} \leftarrow t_{delta}$ / t_0

  w $\leftarrow -\dfrac{\dfrac{n_0}{t_0}}{t_{error}}$

return w

Algorithm 3: WEIGHT Returns the weight of an edge connecting an exit event to an enter event representing likelihood (smaller number → more likely)

than or equal to t_0. The variable t_{error} represents the error rate that the actual travel time is from the average travel time, penalizing a pair the further away the actual travel time is from the average. The two separate calculations for t_{error} ensure equal penalization whether the error is above or below the average. Once I have that calculated, the weight returned is:

$$-\frac{\frac{n_0}{t_0}}{t_{error}}$$

This equation correlates the likelihood of an exit and enter event being linked together to the number of vehicles that previously traveled between the events' locations and the time they took. Thereby, the more vehicles that traveled from x's location to e's location, the more likely they are linked. The rational for this is that higher vehicle counts indicate a main road as opposed to a smaller streets, and if historically many vehicles traveled from x's location to e's location, then it is highly likely that a vehicle leaving x's location also traveled to e's location. Also, the shorter the average time taken to travel from x's location to e's location, the more likely they are linked. The rational behind this is that a longer travel time equates to more time spent in the mix zone, areas outside of the area of observation. This time can be spent changing pseudonyms without the attacker's knowledge. A longer travel time also decreases the magnitude of the weight returned because vehicles normally take the shortest route possible to get from point A to point B. If there are two enter events e_1 and e_2 that are both possibly matched to exit event x_1, and the travel time from x_1 to e_1 is less than the time from x_1 to e_2, then I know that it is possible to get between the two locations in the shorter travel time. Therefore, the shorter travel time is the more likely corresponding enter event and the longer travel time is less likely.

Previous implementations of this weight function use the difference between t_{delta} and t_0 combined with a multiplier as a the numerator: $\frac{n_0}{|t_{delta}-t_0|+0.000001}$. This equation

makes the actual value of the average travel time less signficant, and makes its value relative to the actual travel time. I found that my equation, which gives more credence to the average travel time, while maintaining the penalty of a large difference between the actual and average travel times, gave us better results both in finding DSRC-to-DSRC associations, and in turn WiFi-to-DSRC associations.

Attack Phase

Now that I have explained the weight algorithm, I will explain the attack portion of the attack. This attack was previously used in an online attack [6], but I adapt it to my passive attack by recreating the exit and enter events as they happen by stepping through a sorted list of observation of intervals. First, I sort the set of all observation intervals by ascending time order to recreate the recording of events in the order they actually happened. For each observation interval *obi*, I record *obi*'s start time and location as enter event e and put it into the set of all enter events. I then search within the set of all currently recorded exit events for an exit event with a matching pseudonym to e. If there exists one, then I match them and remove both from their respective sets. I then record the *obi*'s end time and location as the exit event and put it into the set of all exit events.

I then create a matrix to represent the bipartite graph. $Element_{ij}$ represents the weight of the edge between exit event x_i and enter event e_j. I initialize all values to 0. Then I calculate the weight value for every pair of enter and exit events, recording the greatest magnitude weight, excluding magnitudes of MAX_NUM (an extremely large number) which are equivalent to known associations. The maximum weight is used for differentiating between associations and possible associaitons later on. I then obtain a minimum sum graph by using *NumPy* to solve the linear sum problem on the bipartite graph. Iterating through each edge of the solved graph, I extract

Input: dgroups: Dictionary mapping DSRC pseudonym to list of observation intervals, mtx: D2D matrix, mw_perc: percentage of maximum weight

Result: Updates D2D matrix with associations found

learn ← Learning_Phase()

enter_events ← ∅

exit_events ← ∅

Sort observation intervals by ascending time order

foreach *observation interval obi* **do**

    Record enter_event e using $(obi.start, obi.pseudonym, obi.location)$

    if ∃ *exit_event* $x \in$ *exit_events with the same pseudonym as* e **then**

        Remove each from their respective set

    Record exit_event x using (obi.end, obi.pseudonym, obi.location)

end

$x2e$ ← (Number of exit_events) x (Number of enter_events) matrix and initialize all entries to 0

foreach *exit_event* x_i **do**

    foreach *enter_event* e_j **do**

        x2e[i, j] ← $weight(x_i, e_j, learn)$

        Keep track of greatest magnitude weight (excluding MAX_ NUM i.e., known associations)

    end

end

Solve minimum sum graph for $x2e$

foreach *edge ej in solved graph* **do**

  if | *ej's weight*| *is mw_perc of the max weight observed* **then**

    Set ej's events' pseudonyms to associated in mtx;

  else

    Set ej's events' pseudonyms to possibly associated in mtx;

end

Algorithm 4: X2E_Attack Matches exit events to enter events to associate pseudonyms

the pseudonyms from each edge's exit and enter events. If the weight of the edge is above some percentage of the maximum weight, then I set the two pseudonyms to be associated. If not, then I set them to be possibly associated due to the low precision of attack method. By differentiating between the greater weighted edges (i.e., the more likely to be correct) and the lower weighted edges (i.e., the less likely correct), I am able to improve my results while still giving some credence to the lower weighted edges. For my experiments, I use a percentage of 50%. Therefore, edges that have a weight greater than 50% of the maximum recorded weight are considered to be associations, otherwise the edges are labeled as possibly associated. I then create a matrix to represent the bipartite graph. $Element_{ij}$ represents the weight of the edge between exit event x_i and enter event e_j. I initialize all values to 0. Then I calculate the weight value for every pair of enter and exit events, recording the greatest magnitude weight, excluding magnitudes of MAX_NUM which are equivalent to known associations. The maximum weight is used for differentiating between associations and possible associaitons later on. An example of a completed weighted bipartite graph is shown in Figure 3.3.

I then obtain a minimum sum graph by using NumPy to solve the linear sum problem on the bipartite graph. Iterating through each edge of the solved graph, I

extract the pseudonyms from each edge's exit and enter events. If the weight of the edge is above some percentage of the maximum weight, then I set the two pseudonyms to be associated. If not, then I set them to be possibly associated due to the low precision of the attack method. By differentiating between the greater weighted edges (i.e., the more likely to be correct) and the lower weighted edges (i.e., the less likely correct), I am able to improve my results while still giving some credence to the lower weighted edges. For my experiments, I use a percentage of 50%. Therefore, edges that have a weight greater than 50% of the maximum recorded weight are considered to be associations, otherwise the edges are labeled as possibly associated.

4.3.4 Transitive Property

At the end of the D2D matrix construction, I apply the transitive property across the matrix first for associations, then disassociations. For example, if DSRC pseudonym d_1 and d_2 are associated, and d_2 and d_3 are associated, then by the transitive property they are all associated to each other. I do not apply the transitive property for possible associations due to the low confidence of possible associations. Applying the transitive property for D2D possible associations caused a cascading number of incorrect associations when an incorrect possible association bridged different associated groups (groups where all member within a group is associated to each other). Because disassociations are applied after associations, disassociations will take precedence over an association if there is conflicting data.

4.4 WiFi-to-DSRC

After construction of the D2D matrix, it is used in the construction of the W2D matrix. The W2D matrix construction begins by constructing an $N+1$xM matrix where N and M are the number of unique WiFi identifiers and unique DSRC pseudonyms

Input: dgroups: Dictionary mapping DSRC pseudonym to list of observation intervals

Output: MxM matrix representing DSRC-to-DSRC association information, where M = number of DSRC pseudonyms seen

$mtx \leftarrow MxM$ matrix

initialize all values to -1, and diagonal to 1

foreach *DSRC* d_i **do**

  foreach *DSRC* d_j *where there is no information for mtx[i,j]* **do**

    Compare all of d_i's observation intervals to all of d_j's observation intervals

    if *any overlap in time and are at different listeners* **then**

      set d_i and d_j as disassociated

  end

  Same_Listener_Attack()

  X2E_Attack()

  Apply transitive property to mtx for associations and disassociations (NOT possible associations)

  return mtx

end

Algorithm 5: BUILD_D2D_MTX Constructs association matrix of DSRC-to-DSRC associations, -1:no information, 0:disassociation, 1:association, (0,1):possible association

observered across all listeners, respectively. An extra row is added to represent all vehicles in the area of observation that do not have WiFi capbailities, if any exist. Elements in the matrix range in values between 0 and 1 (inclusive) and are representative of the probability that their corresponding WiFi identifier and DSRC pseudonym are associated. Each element is initialized to be $\frac{1}{N+1}$, that is each pseudonym is equally likely to be associated to any identifier. Because an identifier can be associated with many pseudonyms, but a pseudonym can only be associated to a single identifier, each column must sum to 1, while each row can sum to any arbitrary value. After intialization, the matrix construction consists of finding disassociations, known associations, and then possible associations. Possible associations are handled through the use of boosting. There are three separate phases or levels of boosting, with each subsequent level holding less authority than the previous. Therefore, the more likely possible associations are boosted in the first level of boosting, those less likely in the second, and those least likely in the final. Each step of the W2D matrix construction will be explained in the following subsections.

4.4.1 Disassociations

I begin by finding disassociations between identifiers and pseudonyms using Algorithm 6. I do this in a similar manner as when finding DSRC-to-DSRC disassociations, where an identifier and pseudonym are found to be disassociated if they have overlapping time intervals recorded at different listeners, assuming non-overlapping listener coverage. Any identifier-pseudonym pair (w_i, d_j) found to disassociated will have its corresponding element $element_{ij}$ set to 0, indicating that the two elements cannot be associated. $Element_{ij}$'s original value is then evenly distributed among all elements in column j that do not currently equal 0. This is indicative of d_j being found to not be a possible association to w_i. Thus, d_j is more likely to be associated

to any other WiFi identifier that it has not yet been been found to be disassociated from.

Disassociations found from overlapping time intervals at different listeners are extended by using known DSRC-to-DSRC associations from the D2D matrix. For a given WiFi identifier w_i and its set of disassociated DSRC pseudonyms not_assoc, I also conclude that w_i is disassociated from d_k, where d_k is associated to at least one pseudonym in not_assoc. For example, if w_1 and d_1 are found to be disassociated, but d_1 and d_2 were previously found to be associated during the D2D matrix construction, then w_1 and d_2 are also concluded to be disassociated.

Input: mtx: Current W2D matrix, wgroups: Dictionary mapping WiFi identifier to list of observation intervals, Δ: Redistribution factor - value between (0,1)

Output: W2D matrix with found disassociations applied

foreach *WiFi* $w_i \in wgroups$ **do**

  foreach *DSRC* d_j *where there is no information for mtx[i,j]* **do**

    if w_i *and* d_j *have observation intervals* o_i *and* o_j *overlapping in time* **then**

      if o_i *and* o_j *are at different listeners* **then**

        Add d_j to w_i's list of disassociated

        Remove d_j from all of w_i's list of possible associations by location if present

      else

        if d_j *wasn't previously found to be disassociated from* w_i **then**

          Add d_j to w_i's list of possible associations based on location

  end

end

foreach *WiFi* w_i *and its list of disassociated not_assoc* **do**

  not_assoc $\leftarrow$ Union not_assoc with set of all DSRCs that are associated to at least one of the DSRCs in not_assoc

  foreach *DSRC* d_j *in not_assoc* **do**

    Set w_i and d_j to be disassociated in mtx

    Distribute the original value in mtx equally among all other rows in column i that don't currently have a value 0

  end

end

Algorithm 6: W2D_DISASSOCIATIONS Finds and applies W2D disassociations into the W2D matrix

4.4.2 Known Associations

I then find known associations between identifiers and pseudonyms. Known associations are associations with such high confidence that the attack model automatically concludes that they are actual associations. An identifier and pseudonym pair are found to be known associated if they have overlapping observation intervals and no other signals are seen at the same time, that is, during the time the identfier's observation interval and pseudonym's observation interval overlap, there are no other overlapping observation intervals. An example of this is shown in Figure 1.4, where w_1 and d_1 are both seen at a listener at a some point in time with no other signals around. Any known associated identifier-pseudonym pair (w_i, d_j) will have its corresponding element $element_{ij}$ set to 1 and all other elements in column j are set to 0 to maintain the summation of 1. Algorithm 7 contains the algorithm for both finding known associations and applying Level 1 boosting due to their similar criteria.

Input: mtx: Current W2D matrix, wgroups: Dictionary mapping WiFi identifier to list of observation intervals, Δ: Redistribution factor - value between (0,1)

Output: W2D matrix with found associations and Level 1 possible associations applied

$d2w \leftarrow emptyhashmap$;

foreach *WiFi* w_i *and its list of lists of possibly associated DSRCs by location* *assocs_ by_loc* **do**

  foreach *list of possibly associated DSRCS assocs* $\in$ *assoc_by_loc* **do**

    if *only 1 DSRC* $d_j \in$ *assocs* **then**

      Record that d_j was seen with $w_i + 1$ using $d2w$;

      $mtx[i, j] \leftarrow 1$;

      $mtx[i', j] \leftarrow 0$, where i' corresponds to each WiFi identifiers excluding w_i;

      $mtx[i, j'] \leftarrow 0$;

      where j' corresponds to all DSRC pseudonyms disassociated to d_j;

      Redistribute each mtx[i,j'] value evenly among all other rows in column j' that don't equal 0;

    end

    else

      foreach *DSRC* $d_j in$ *assocs* **do**

        Record in $d2w$ that d_j was seen with $w_i + 1$;

      end

    end

  end

end

foreach *DSRC* d_j *in d2w* **do**

  $not\_rs \leftarrow$ all WiFi identifiers not seen with d_j;

  $\Sigma \leftarrow$ sum of probabilites of identifiers in not_rs;

  $total \leftarrow$ total number of times d_j was seen with any WiFi;

  foreach *WiFi* w_i *in d2w*$[d_j]$ **do**

    $mult \leftarrow$ number of times d_j was seen with w_i;

    mtx[i,j] += $\frac{\Delta * \Sigma}{total}$ * $mult$;

  end

end

Algorithm 7: W2D_KNOWN ASSOCIATIONS_AND_LEVEL1_BOOSTING Finds and applies W2D associations and Level 1 possible associations into the W2D matrix

4.4.3 Possible Associations

I then find possible associations between identifiers and pseudonyms. An identifier and pseudonym pair are found to be possibly associated if they have overlapping observation intervals, but there are also other overlapping intervals around belonging to other pseudonyms. An example of this is shown in Figure 1.5, where WiFi identifier w_1 is seen with both DSRC pseudonyms d_1 and d_2. Because w_1 is seen with both d_1 and d_2 at the same listener at the same point in time, then I say that w_1 is possibly associated to both d_1 and d_2. Possible associations are incorporated into the matrix through the use of boosting, and because boosting is done independently in each column (i.e., boosting is done in terms of DSRC pseudonyms), I need to invert the mapping of WiFi identifiers to their list of possibly associated DSRC pseudonyms into DSRC pseudonyms to their list of possibly associated WiFi identifiers. I do this by iterating through each WiFi identifier w_i's list of DSRC pseudonyms *poss_assocs*. For each DSRC pseudonym d_j in *poss_assocs*, I record that w_i was seen with d_j, keeping a count of the number of times each pair is seen together. This mapping is

used during boosting to figure out which WiFi identifiers are being boosted for each DSRC pseudonym.

There are three levels of possible associations, each corresponding to a level of boosting. Because the boosted amount is based on the summation of those less likely to be associated to a given pseudonym, each subsequent level of boosting is less magntudinous than preceding levels. I also use different Δs (redistribution factors) at each level of boosting. While these Δ values can be tuned using machine learning techniques to get the optimal values, I choose my values based on experimentation. I explain my reasoning for my chosen values in the following subsections, and present the effects of changing the Δ values has on my attack results in Chapter 5. Level 1 boosting handles immediate possible associations, where for a given pseudonym d_j, I boost identifiers seen directly with d_j. Level 2 boosting handles possible associations of a single degree of separation, where for a given pseudonym d_j, I boost identifiers seen with a pseudonym that is associated to d_j. And Level 3 boosting handles possible associations with a single "weak" degree of separation, where for a given pseudonym d_j, I boost identifiers seen with a pseudonym that is possibly associated to d_j.

Level 1 Boosting

The first level of boosting handles the immediate possible associations. That is, a DSRC pseudonym d_j and WiFi identifier w_i are said to be possibly associated if w_i and d_j have a pair of overlapping oberservation intervals, but there were other pseudonyms around at the time. For each DSRC pseudonym d_j and its set of WiFi identifiers it was seen with $seen\_with$, I calculate Σ (the total probability of the lesser likely to be associated identifiers) using the set of identifiers:

$$\{w \mid \text{w is an identifier less likely to be associated to } d_j\} =$$
$$\{w \mid \text{w is an identifier seen by a listener}\}$$
$$-\{w \mid \text{w is an identifier overlapping with } d_j \text{ with other pseudonyms around}\} \tag{4.1}$$

At Level 1, I use a Δ of 0.6. From my experiments, I saw that 0.6 gave us a relatively high number of true positives, without dramatically increasing the number of false positives. I show the effects of changing the Δ value on my attack results in Chapter Chapter 5. For each of the less likely associated identifiers, I multiply their corresponding elements by $(1 - \Delta)$; making the amount to be redistributed among the possible associations to be $\Sigma * \Delta$. To begin redistribution for a given DSRC pseudonym d_j, I sum up the total number of times it was seen with any identifier ($total\_seen\_with$), then for each identifier w_i to be boosted, I boost w_i by

$$\frac{\Sigma * \Delta}{total\_seen\_with} * mult_i$$

where $mult_j$ is the number of times w_i was seen with d_j. This allows us to favor WiFi-DSRC pairs that are seen more often together by giving them a larger boost. For example, if identifier w_1 and pseudonym d_1 are seen together twice and w_2 and d_1 are seen together only once, then the d_1 should be more likely to be associated to w_1 than w_2.

Level 2 Boosting

The second level of boosting handles possible associations by a degree of separation based on known DSRC-to-DSRC associations. That is, a DSRC pseudonym d_j and WiFi identifier w_i are said to be possibly associated if w_i and d_k have a pair of overlapping oberservation intervals, but there were other pseudonyms around at the

time, where d_k is any pseudonym associated to d_j. For Level 2 boosting, the set of identifiers that I take the boost from is:

$$\begin{aligned}
&\{w \mid \text{w is an identifier less likely to be associated to } d_j\} = \\
&\{w \mid \text{w is an identifier seen by a listener}\} \\
&- \{w \mid \text{w is an identifier overlapping with } d_j \text{ with other pseudonyms around}\} \\
&- \{w \mid \text{w is an identifier overlapping with a pseudonym associated to } d_j \text{ with others around}\}
\end{aligned} \tag{4.2}$$

Note that for a given pseudonym d_j, I remove both identifiers seen with a pseudonym associated to d_j and identifiers seen with d_j from the set of identifiers that I take the boost from. This is to prevent penalizing any identifiers that were boosted in Level 1.

At Level 2, I use the same boosting method as Level 1, but a different Δ value, Δ', that is 1.5x the original Δ value, with a max of 0.99. In my experiments, I use a value of 0.9 ($= 0.6 \cdot 1.5$). I choose a higher Δ' than Δ because the possible associations being boosted at Level 2 are near the same precision as Level 1 possible associations. They are of a near precision due to being only a single degree of separation and the high precision of my DSRC-to-DSRC associations. Remember that the boosted amount is determined by both the redistribution factor Δ' and Σ, and Σ decreases each subsequent boosting level due to both the set of identifiers that I take the boost from growing smaller and their corresponding matrix element probability values decreasing if previous levels of boosting occurred. Therefore, due to a potentially smaller Σ value, to maintain a near equal boosting effect at Level 2 as Level 1, I increase the redistribution factor. For example, let's say w_1 is boosted in Level 1 and w_2 is boosted in Level 2 for a given pseudonym. If $\Sigma = 0.9$ during Level 1 boosting and $\Delta = 0.6$, then w_1 gets a boost of 0.54 ($0.6 \cdot 0.9$), and afterwards each element that constituted

the Σ calculation is reduced by the Δ in terms of percentage. Therefore, during Level 2 boosting, now $\Sigma = 0.36$ $(= 0.9 \cdot (1 - 0.6))$. Using the redistribution factor $\Delta' = 0.9$, w_2 gets a boost of 0.324 $(= 0.9 \cdot 0.36)$. While not the exact same boost, the Level 2 boost was similar to the Level 1 boost, with the difference being argued that the precision of the D2D associations is not 100%. Therefore, Level 2 boosting has a proximate level of confidence as Level 1, but not the same. If a lower Δ' value had been chosen, then the boosted amount would have been even smaller, which can be useful if new methods are integrated into the D2D matrix construction that increase recall, but lower precision.

Aside from boosting possible associations, I also make known associations in this phase, with associations having higher precedence than boosting. If a pseudonym is found to have a known association with an identifier, then the association is automatically recorded in the W2D matrix and none of its possible associations at this level are boosted. I apply the same logic of a single degree of separation when looking for associations at this step. An identifier-pseudonym pair w_i and d_k are found to be associated if w_i and d_k have overlapping intervals and no other pseudonyms were around at the time, where d_k is any pseudonym associated to d_j. In other words, if an identifier w_i was found to be associated to d_k, then any pseudonym associated to d_j is also associated to w_i. Known associations can be made using the single degree of separation due to the high precision of the D2D association matrix. If the D2D precision were to substantially drop, then these known associations using a single degree of separation would not be as effective and often create incorrect known associations.

Input: mtx: Current W2D matrix, wgroups: Dictionary mapping WiFi identifier to list of observation intervals, Δ: Redistribution factor - value between (0,1)

Output: W2D matrix with found Level 2 possible associations applied

d2w_t $\leftarrow$ *emptyhashmap*;

foreach *DSRC* d_j **do**

  foreach *DSRC* d_k *that is associated to* d_j *excluding* $d_k = d_j$ **do**

    if d_j *was ever only seen with 1 WiFi (i.e., automatic association* **then**

      $mtx[i, k] \leftarrow 1$;

      $mtx[i', k] \leftarrow 0$, where i' corresponds to each WiFi identifiers excluding w_i;

      $mtx[i, k'] \leftarrow 0$;

      where k' corresponds to all DSRC pseudonyms disassociated to d_j;

      Redistribute each mtx[i,k'] value evenly among all other rows in the k' column that don't equal 0;

      Remove d_j from $d2w\_t$ and break;

    else

      Record in $d2w\_t$ that d_j is possibly associated with each WiFi identifiers w_i seen with d_k for the number of times d_k and w_i were seen together;

  end

end

$\Delta' \leftarrow \min(1.5 * \Delta, .99)$;

foreach *DSRC* d_j *in d2w_t* **do**

$not\_rs \leftarrow$ [all WiFi identifiers] $\cap$ [WiFi identifiers seen with d_j] $\cap$ [WiFi identifiers seen with a DSRC associated to d_j];

$\Sigma \leftarrow$ sum of probabilites of identifiers in not_rs;

$total \leftarrow$ total number of times a DSRC associated to d_j was seen with any WiFi;

foreach *WiFi* w_i *in d2w_t* $[d_j]$ **do**

$mult \leftarrow$ number of times a DSRC associated to d_j was seen with w_i;

mtx[i,j] += $\frac{\Sigma * \Delta'}{total}$ * $mult$;

end

end

Algorithm 8: W2D_LEVEL2_BOOSTING Finds and applies W2D Level 2 possible associations into the W2D matrix

Level 3 Boosting

The third and final level of boosting handles possible associations by a degree of separation based on possible DSRC-to-DSRC associations. That is, a DSRC pseudonym d_j and WiFi identifier w_i are said to be possibly associated if w_i and d_k have a pair of overlapping oberservation intervals, but there were other pseudonyms around at the time, where d_k is any pseudonym possibly associated to d_j. For Level 3 boosting, the set of identifiers that I take the boost from is:

$\{w \mid \text{w is an identifier less likely to be associated to } d_j\} =$

$\{w \mid \text{w is an identifier seen by a listener}\}$

$- \{w \mid \text{w is an identifier overlapping with } d_j \text{ with other pseudonyms around}\}$

$- \{w \mid \text{w is an identifier overlapping with a pseudonym associated to } d_j \text{ with others around}\}$

$- \{w \mid \text{w is an identifier overlapping with a pseudonym possibly associated to } d_j \text{ with others around}\}$

(4.3)

At Level 3, I use the same boosting method as previous levels, but again a different Δ value, Δ'', that is .75x the original Δ value. In my experimenets, I use a value of 0.45 ($= 0.6 \cdot 0.75$). I choose a lower Δ'' value than Δ due to the high uncertainty of DSRC-to-DSRC possible associations. I saw from my experiments that if I used too high of a Δ'' value, then that created too many false positives, with very few true positives. Remember that these possible associations are found from exit-enter event pairs from the X2E Attack with a weighted edge that did not meet a weight value threshold. While these possible associations have a high undercertainty, I still wanted to give them some credence, so I use a low value redistribution factor in my Level 3 boosting. I also do not extend the automatic known associations used in the previous phase to this phase. Similarly to the low value Δ'', I do not extend known associations in this phase due to the high uncertainty of these possible associations, where if I did extend known associations, then the number of false positives made would greatly outnumber any true positives made.

Input: mtx: Current W2D matrix, wgroups: Dictionary mapping WiFi identifier to list of observation intervals, Δ: Redistribution factor - value between (0,1)

Output: W2D matrix with found Level 3 possible associations applied

d2w_mb *getsemptyhashmap*;

foreach *DSRC* d_j **do**

  foreach *DSRC* d_k *that is possibly associated to* d_j *excluding* $d_k = d_j$ **do**

    Record in d2w_mb that d_j is possibly associated with each WiFi identifiers w_i seen with d_k for the number of times d_k and w_i were seen together;

  end

end

$\Delta'' \leftarrow .75 * \Delta$;

foreach *DSRC* d_j *in d2w_t* **do**

  $not\_rs \leftarrow$ [all WiFi identifiers] $\cap$ [WiFi identifiers seen with d_j] $\cap$ [WiFi identifiers seen with a DSRC associated to d_j] $\cap$ [WiFi identifiers seen with a DSRC possibly associated to d_j;

  $\Sigma \leftarrow$ sum of probabilites of identifiers in not_rs;

  $total \leftarrow$ total number of times a DSRC possibly associated to d_j was seen with any WiFi;

  foreach *WiFi* w_i *in d2w_mb* $[d_j]$ **do**

    $mult \leftarrow$ number of times a DSRC possibly associated to d_j was seen with w_i;

    mtx[i,j] += $\frac{\Sigma * \Delta''}{total}$ * $mult$;

  end

end

Algorithm 9: W2D_LEVEL3_BOOSTING Finds and applies W2D Level 3 possible associations into the W2D matrix

4.4.4 WiFi Null Boosting

The final step of the W2D matrix construction is WiFi Null Boosting, where I apply the boosting method to the WiFi Null row for any DSRC pseudonym seen without an overlapping WiFi identifier. I do this by keeping a count for each DSRC pseudonym of the number of times it was seen without any overlapping WiFi identifier. Iterating through each listener's recorded observation intervals, if a listener l did not see any WiFi signals, then for each DSRC pseudonym that l recorded an observation interval for, I increment its count by 1. Once I have calculated each pseudonym's count, for each pseudonym d_j, I boost WiFi Null the number of times d_j was seen without an overlapping WiFi signal. I use a redistribution factor value of 0.1 for my experiments.

The reasoning for this approach is that DSRC pseudonyms originating from vehicles without WiFi capability would be more likely to be seen without an overlapping WiFi signal than pseudonyms originating from vehicles with WiFi capability. However, due to the nature of the signal protocols' heart rates, many more pseudonyms are seen without an overlapping WiFi signal than those that actually originate from a vehicle without WiFi capaiblity. Due to the high uncertainty of this method, I opted to use a low redistribution factor as I still wanted to factor in pseudonyms seen without a WiFi signal, but not detract from the findings of other methods in the W2D matrix construction. The pseudocode for WiFi Null Boosting is found in Algorithm 10.

The WiFi signal in my experiements is acting as a base station, where it broadcast to announce its presence. In a real-world scenario, this can be affected by cellular devices that act as WiFi hotspots. For the purposes of this project, I assume that vehicles are the only source of WiFi signals within the areas.

Input: mtx: current (N+1)xM matrix for WiFi-to-DSRC associations, l2wobis: Dictionary mapping listener to its set of WiFi observation intervals, l2dobis: Dictionary mapping listener to its set of DSRC observation intervals, Δ_{null} : Redistribution factor for WiFi null - value between (0,1)

Result: Boosts DSRC pseudonyms that likely originate from vehicle without WiFi capabilty with WiFi Null row in matrix for the number of times they were seen without an overlapping WiFi signal

d_count $\leftarrow$ *emptyhashmap*

foreach *listener L and its set of DSRC observation intervals dobis* **do**

  if *L has no WiFi observations* **then**

    Increment the count of each DSRC dobis by 1 in d_count

  else

    foreach *DSRC observation interval dobi* $\in$ *dobis* **do**

      if *dobi does not overlap in time with any of L's WiFi observation intervals* **then**

        Increment the count of *dobi*'s DSRC pseudonym by 1 in d_count

    end

end

foreach *DSRC pseudonym d_j* $\in$ *and its count* $\in$ *d_count* **do**

  Boost the WiFi_Null probability *count* times in mtx using Δ_{null} as the redistribution factor, taking the boost from all rows except WiFi_Null

end

Algorithm 10: WiFi_Null_Boost Boosts WiFi Null probability for DSRC pseudonyms possibly originating from vehicle without WiFi capability

4.5 Confidence Threshold

Once the W2D matrix construction is completed, I use a confidence threshold to conclude which of the associations I actually consider to be associations. For my experiments, I use a confidence threshold of 0.5. That is, if an $element_{ij}$ of the W2D matrix is greater than 0.5, then I conclude that WiFi identifier w_i and DSRC pseudonym d_j are associated. I use a threshold of 0.5 because any probability over 50% indicates a majority. Remember that each column sum to 1 or 100%. Therefore, if I used a threshold under 50%, then there is a likelihood that a pseudonym is found to be associated to multiple identifiers. Therefore, 0.5 is the minimum threshold required to ensure an pseudonym is associated to only a single identifier.

4.6 Path Reconstruction

I then use the W2D associations to reconstruct a vehicle's path. For each WiFi identifier w_i, I find all DSRC pseudonyms d_j in set D such that $element_{ij}$ in the W2D matrix meets the confidence threshold. All observation intervals with a pseudonym belonging in D or the identifier w_i are ordered in ascending time order, and a Euclidian distance is drawn between subsequent intervals' listeners. Because the main concern of this thesis was pseudonym association and not path reconstruction, I opted for this simplistic approach. A more accurate path can be reconstructed by using a Manhattan distance line or the actual target map's roads to connect subsequent listeners.

Chapter 5

EXPERIMENTS AND RESULTS

I use the SUMO traffic simulator [2] to simulate realistic vehicle traffic in three cities: Eichstätt, Mions, and Melun. I choose these towns, a German town and two French communes, because they have centralized medium-size downtown districts that would realistically see a lot of vehicle traffic. The Eichstätt map measures approximately 12 km^2 in area and its traffic simulation had 200 vehicles with 1627 pseudonyms across all vehicles, and average trip length of 4.5 km. The Mions map measures approximately 5.7 km^2 in area and its simulation had 300 vehicles with 1515 pseudonyms, and an average trip length of 1.8 km. The Melun map measures approximately 8.7 km^2 in area and its simulation had 300 vehicles with 2284 pseudonyms, and an average trip length of 2.4 km.

Because I do not have complete listener coverage, I am unable to receive a packet from every identifier and pseudonym. Therefore, I evaluate my attack solely based on the packets received by the listeners. In my simulations, I use simulated DSRC-like and WiFi-like radios as the long-distance pseudonym changing protocol and short-distance non-changing protocol, respectively. The different parameters for my radio protocols are summarized in Table 5.1. For my simulations, the DSRC protocol has a range of 100 meters, a heart rate of 0.1 seconds, and a change rate of 30 seconds; the WiFi protocol has a range of 25 meters and a heart rate of 30 seconds. Because WiFi uses a non-changing identifier, I set its change rate to ∞.

Table 5.1: Radio Protocol Parameters for Simulations

Radio	range (m)	heart rate (s)	change rate (s)
DSRC	1000	0.1	30
WiFi	25	30	∞

To model packet reception at the listeners, I step through each simulated vehicle's path, stopping at the given heart rates of the signals to broadcast the respective signals. I use varying WiFi penetration rates, ranging from 50% to 100% while maintaining a 100% penetration rate for DSRC capability. The WiFi penetration rate defines what percentage of vehicles within the network have WiFi capability. I use varying WiFi penetration rates to test the robustness of the attack when not all vehicles have WiFi capability. Non WiFi enabled vehicles will continue to emit DSRC packets. An attack will not know that pseudonyms associated with these packets originate from a vehicle without WiFi capability, and thus varying WiFi penetration rates test the ability of the attack to filter out *noise* pseudonyms. I assume a 100% DSRC penetration rate due to its many appplications in road safety. Therefore, it is likely to be federally mandated and required in future modern vehicles. Signal broadcast data is stored in a MongoDB database; I record the broadcasting vehicle's ID, the coordinate of the broadcast, the time of broadcast, the signal protocol, and the pseudonym or identifier associated to the signal at the time. I use a handpicked listener placement setup for each map, with a set of 20 listeners and a focus on heavy traffic areas, such as busy intersections. During listener placement, I also ensure that listeners do not overlap in coverage, as many of my attack methods assume non-overlapping listener coverage. Example listener placement configurations for each map are illustrated in Figures 5.1, 5.2, and 5.3, where a red circle represents the coverage of a single listener. I implement my attack in Python with the usage of the NumPy package. Because my attack model is a generalizable model, the following parameters can be configured to simulate a variety of scenarios:

- Map and listener placements
- Long-distance pseudonym changing protocol
  - range
  - heart rate

  - change rate
- Short-distance non-changing protocol
  - range
  - heart rate
  - penetration rate
- Maximum Weight percentage for X2E Attack
- D2D Matrix Construction steps
- Δ, Δ', Δ'' (Redistribution factors for Boosting in W2D matrix construction)
- Confidence Threshold

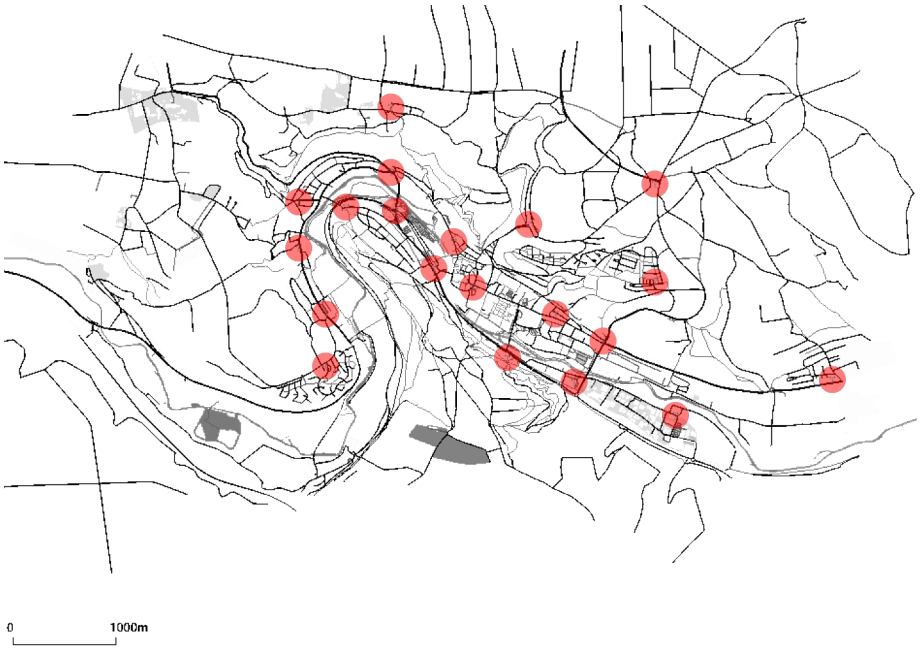


Figure 5.1: Example SUMO map with listener placements, Map: Eichstätt

Figure 5.2: Example SUMO map with listener placements, Map: Mions

Figure 5.3: Example SUMO map with listener placements, Map: Melun

To measure the effectiveness of my results, I measure the number of associations made, the number of correct associations made, the average W2D (WiFi-to-DSRC) association precision, the average W2D association recall, the total D2D (DSRC-

to-DSRC) association precision, and the total D2D association recall. Precision is a measure of how accurate the average guessed association is. That is, among the guessed associations, how many of them were correct. Precision is equal to the number of true-positives divided by the sum of true-positives and false-positives, that is the number of correct associations made divided by the total number of associations made. Recall is a measure of how many of the total actual associations we got correct. Recall is the number of true-positives divided by the sum of true-positives and true-negatives, that is the number of correct associations made divided by the total number of actual associations.

$$\begin{aligned} Precision &= \frac{TruePositive}{TruePositive + FalsePositive} \\ &= \frac{\text{Correct Associations Made}}{\text{Total Guessed Associations}} \end{aligned} \tag{5.1}$$

$$\begin{aligned} Recall &= \frac{TruePositive}{TruePositive + TrueNegatives} \\ &= \frac{\text{Correct Associations Made}}{\text{Total Actual Associations}} \end{aligned} \tag{5.2}$$

D2D association precision and recall allow me to gauge how effective my D2D matrix construction strategies are. To calculate these values, I construct a separate D2D association matrix containing all the actual D2D associations involving pseudonyms seen by the listeners. Then I compare my D2D association matrix against the newly constructed actual association matrix; I record the number of true positives, true negatives, and false positives to use in the precision and recall calculations. Making correct D2D associations is fundamentally the goal of previous attacks that are limited to a single signal protocol [6][3][5][24]. However, D2D precision and recall only measure a portion of the effectiveness of the D2D associations. Because I use two signals, the other being the unique identifier of vehicles, the average W2D association precision and recall are also a measure of how useful the D2D association data is when

constructing the W2D matrix. Note that I do not use D2D possible associations or disassociations for any of the precision and recall calculations.

The number of total associations made and number of correct associations is a simple means of measuring the effectiveness of the attack model. It allows me to quickly see the effects of tuning a parameter or changing part of my matrices construction strategies. However, because I am measuring W2D associations and the number of learnable W2D associations changes based on the WiFi penetration rate, these numbers on their own do not tell me much information without a number to compare to. Therefore, I need to also take into account the number of actual associations. The number of learnable W2D associations is based on the WiFi penetration rate because I leverage WiFi identifiers to uniquely identify vehicles; vehicles without WiFi capability are unknown to the attack model. Therefore, the model has no means of associating DSRC pseudonyms to a vehicle without WiFi capaiblity.

The average W2D association precision and recall are the core measurement of how effective the attack model is. For each uniquely identifiable vehicle (i.e. each WiFi identifier), I calculate the precision and recall for that vehicle, then average the calculations across all vehicles to get an average precision and recall.

The following sections detail the results of my experiments running the attack against different scenarios and configurations.

- General Attack Results
  - Measures the overall effectiveness of the attack to create W2D associations
- Attack Across Different Maps
  - Measures the effectiveness of the attack across different maps
- D2D Techniques
  - Measures the effectiveness of different D2D association techniques and how those affect W2D associations

- Δ Tuning
  - Illustrates how different Δ values affect W2D associations made
- Weight Calculations
  - Compares the effectiveness of the attack using my weight calculation in the X2E attack and using the previous weight calculation

5.1 General Attack Results

To show the general effectiveness of the attack model, I run the attack on a single map with WiFi penetration rates ranging from 50-100% in 10% increments to get a measure of the general effectiveness of the attack. For this experiment, I use the traffic simulation on the Eichstätt map, all steps of the D2D matrix construction, $\Delta = 0.6$, $\Delta' = 0.9$, and $\Delta'' = 0.45$.

I see a D2D association precision of about 96% and a D2D association recall of 18%. Comparitively, in a previous paper [6], using their version of the X2E Attack, the authors are only able to correctly match about 6% of the exit-enter trips when the change rate of vehicles is 30 seconds. Their simulation used 250 vehicles with an average trip duration of 479 seconds and 5 observation nodes. My simulation used 20 observation nodes and 200 vehicles with trips ranging between a minute to 9 minutes and an average distance of 4.5 km. While their attack does not evaluate effectiveness in the same measures my attack model does, it does give me a starting point to compare to. Note that their results include any exit-enter event pairs that have the same pseudonyms, something that is not considered in my measurements. In another paper [24], using the contents of the DSRC packets to build anonymous location profiles and associate vehicles to their respective profiles, the authors were able to track vehicles for the entire 1000 second trip duration if vehicles kept their pseudonyms for longer than 3 seconds. However, their techniques required complete

listener coverage and the ability to gather vehicle location and trajectory data from the DSRC packets. When they introduce random spatial noise within the data packets, even a single meter of noise added caused tracking to drop to less than 20% of the trip when there were 250 vehicles present. When even more noise was added (2 to 5 meters), tracking dropped to less than 10% of the trip when using 250 vehicles. They also consider penetration rate into their system, but unlike my simulation model, they consider it as advantageous, as they have less vehicles to track. This second paper exemplifies the tracking capabilities given perfect data gathering capabilities. However, my attack model assumes only partial gathering of information. I use 20 handpicked listeners focusing on high traffic areas, such as busy intersections, but the attack model can be configured to accept any number of listeners and placement configuration.

I see a general decline in both the number of associations made and the number of correct associations made as the WiFi penetration rate decrease, as shown in Figure 5.4. This is to be expected, as the number of actual associations is directly correlated to the WiFi penetration rate.

The average W2D association precision is fairly consistent across all WiFi penetration rates, with an average of about 78.7% (low of 74.8% at the 50% WiFi penetration rate, and a high of 81% at the 70% WiFi penetration rate). The average W2D recall is also fairly consistent across all WiFi penetration rates, with an average of about 21.7% (low of 20.2% at the 100% WiFi penetration rate, and a high of 23.8% at the 70% WiFi penetration rate). These results are shown in Figure 5.5 and show that my attack is consistent across different WiFi penetration rates. As the WiFi penetration rate decreases, the ratio of number of vehicles with WiFi capability and those without begins to skew toward those without. That is, the number of vehicles with WiFi capability decreases, and the number of vehicles without WiFi capabiity increases. Less vehicles with WiFi capaiblity means less number of actual associations to cor-

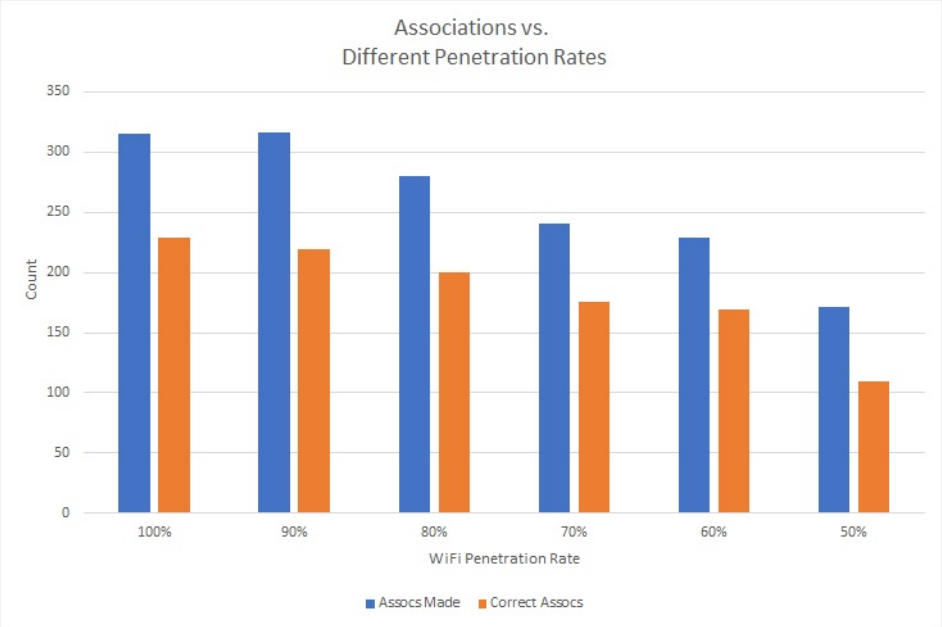


Figure 5.4: Associations vs. Different Penetration Rates

rectly guess, while more vehicles withouth WiFi capability means more DSRC noise in the system, as these vehicles continue to emit DSRC packets. DSRC pseudonyms originating from vehicles withouth WiFi capaiblity do not associate to any identifier known to my system, but are still picked up by listeners and introduce complexity to the system. However, my attack model has shown to perform at the same level of average precision and recall regardless of WiFi penetration rate. This means that, regardless of the WiFi penetration rate, my attack model is able to, for the average vehicle, correctly guess an association 78.7% of the time and find about 21.7% of its total DSRC pseudonyms.

5.2 Attack Across Different Maps

In this section, I test my attack model against different maps to ensure that my attack is effective across not only my initial test map, but also other maps. The traffic simulation scenarios differ among all maps to ensure variance in my tests.

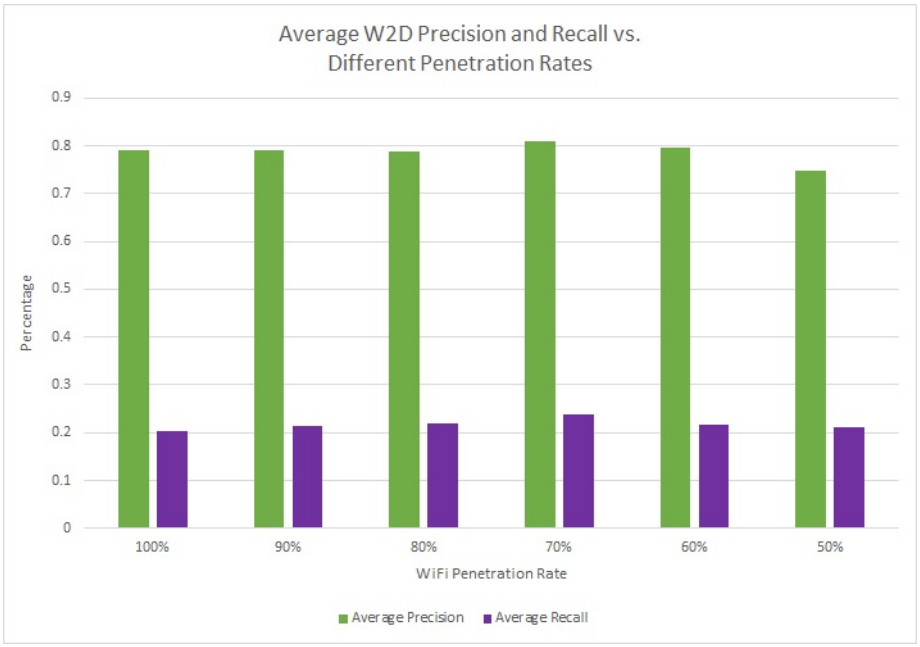


Figure 5.5: Average W2D Precision & Recall vs. Different Penetration Rates

These differences include number of vehicles and number of DSRC pseudonyms. For these tests, I use alternating WiFi penetration rates from 50-100% in 10% increments, all steps of the D2D matrix construction, and Δ values of $\Delta = 0.6$, $\Delta' = 0.9$, and $\Delta'' = 0.45$.

For Melun, I get a D2D precision of 98% and a D2D recall of 24%. For Mions, I get a D2D precision of 98% and a recall of 38.7%. For the orginal map of Eichstätt, I get a D2D precision of 96.7% and a D2D recall of 18.4%. While D2D precision is consistently high across the different maps, D2D recall is not. This may be explained by a number of things. One possiblity is the inclusion of the X2E attack, which is affected by many factors outside of my control. My attack may not have seen every exit and enter event to construct a perfect bipartite graph. If an exit or enter event has a corresponding event that was not seen by one of my listeners, then the event still needs to be paired with another event to construct a minimum sum graph, causing incorrect edges to be chosen. The different traffic simulations also use different number

of vehicles and pseudonyms. Melun had 300 vehicles with 2284 total pseudonyms, averaging around 7 pseudonyms a vehicle. Mions had 300 vehicles with 1515 total pseudonyms, averaging 5 pseudonyms a vehicle. Eichstätt had 200 vehicles with 1627 total pseudonyms, averaging 8 pseudonyms a vehicle. Note that the listeners are unable to pick up a signal packet from every pseudonym, therefore my attack model is only aware of the information received recorded by the listener. In the future, I plan to research a correlation between number of vehicles, total number of pseudonyms, number of pseudonyms per vehicle, or a combination of these factors with my attack results, but this is an area for future research.

Map size also affects the attack model's effectiveness. Because I maintain a set of 20 listeners, the percentage of map coverage is significantly different between the maps. We can see that the Eichstätt simulation, which had the largest map size of 12 km^2, saw the smallest D2D recall of 18.4%, while the Mions simulation, which had the smallest map size of 5.7 km^2, saw the largest D2D recall of 38.7%. A greater listener coverage results in a larger percentage of the radio packets intercepted, that is, more data to form conclusions from. Future research is to explore the correlation between map size and listener density to the effectiveness of the attack model.

Figure 5.6 shows the average W2D precision and recall across all three maps. Across all maps, I get an average precision of about 80% and an average recall of about 23%. For Melun, I get an average W2D precision of 81% and an average W2D recall of 23%. For Mions, I get an average W2D precision of 82% and an average W2D recall of 26%. These are very similar to the results of my initial test on Eichstätt, which saw an average W2D precision of 78% and an average W2D recall of 21%.

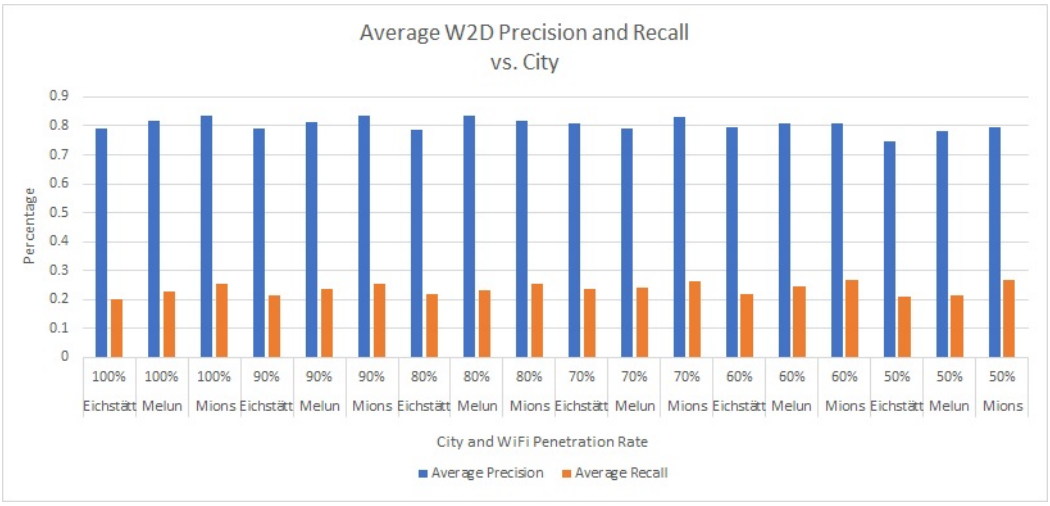


Figure 5.6: Average W2D Precision & Recall vs. Different Maps

5.3 Delta Tuning

In this section, I detail the results of my Δ tuning through experimentation and show the effects of increasing or lowering Δ. While the attack model is not confined to having the redistribution factors for Level 2 and Level 3 boosting (Δ' and Δ'', respectively) be equationally based on Δ, I maintain the use of:

$$\Delta' = 1.5 * \Delta \tag{5.3}$$

$$\Delta'' = 0.75 * \Delta \tag{5.4}$$

to remain consistent. These relative values were chosen as explained in Section 4.4.3. These tests were run on the Eichstätt simulation, using all parts of the D2D matrix construction, and using alternating WiFi penetration rates from 50-100% in 10% increments. The results of my experiements for monitoring the effects of changing the initial Δ value on W2D associations are shown in Figure 5.7 and Figure 5.8.

I see that by increasing Δ from 0.6 up to 0.8, I greatly increase the number of W2D associations made, with only a small percentage of them being correct. This is illustrated in both the raw numbers in Figure 5.7 as well as the relatively similar

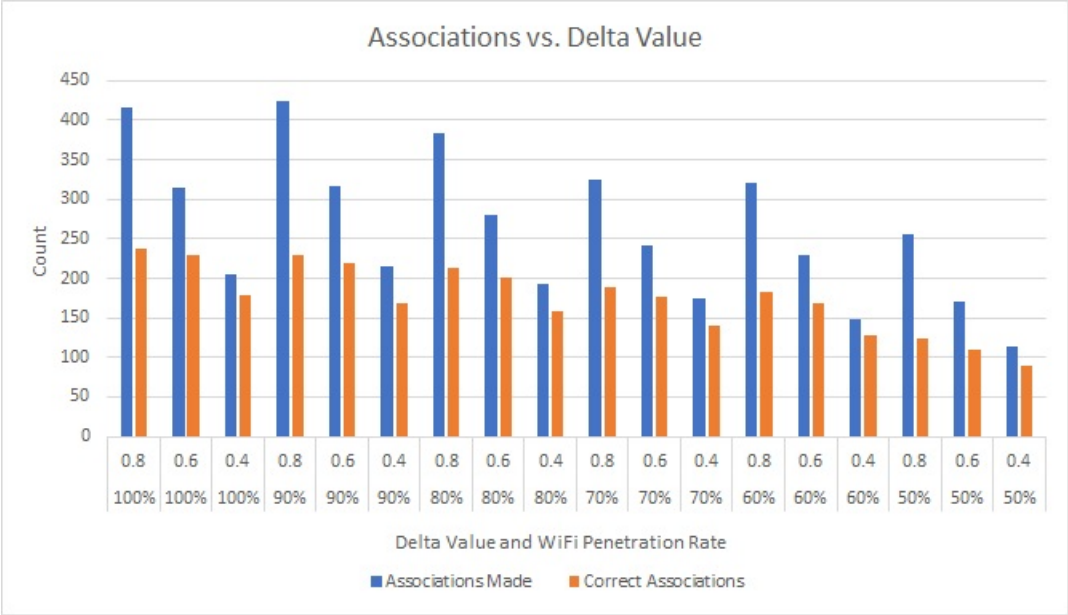


Figure 5.7: Associations vs Delta Value

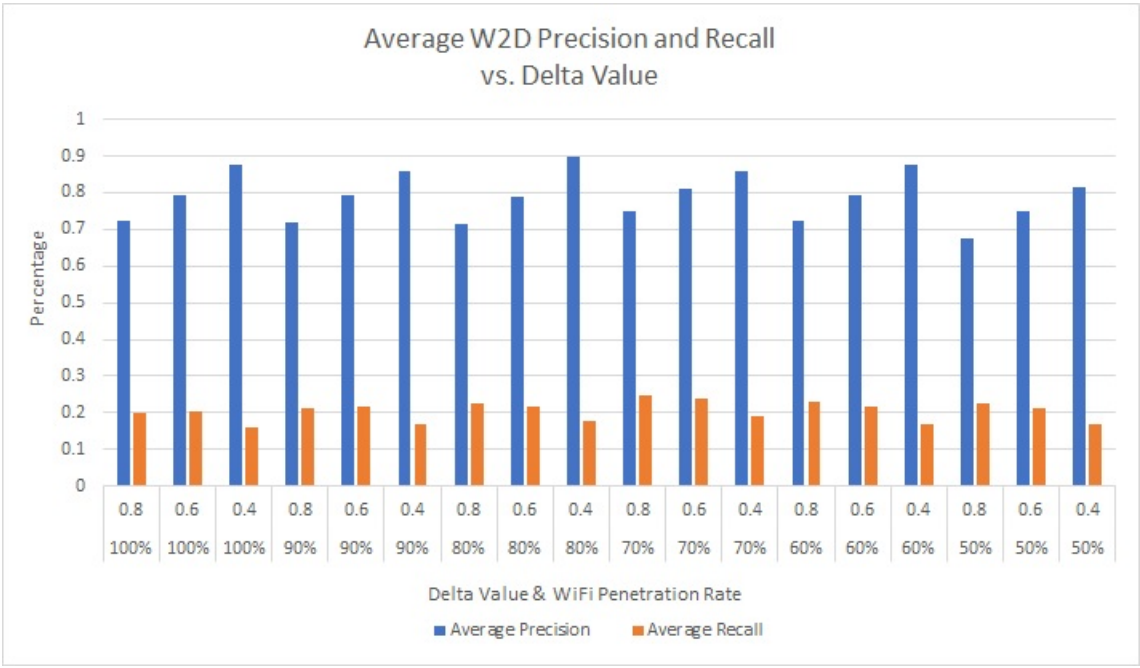


Figure 5.8: Average W2D Precision & Recall vs. Delta Value

average W2D recall values between the two for each WiFi penetration rate, but lower average W2D precision values when $\Delta = 0.8$ as shown in Figure 5.8. The large Δ caused many probabilities that previously were not great enough to meet the confidence threshold to be pushed over the edge. While there is a small increase in average recall (i.e. number of correct associations), the hit to precision caused by the increased number of false positives outweighed the benefits getting a very slightly higher recall from a few more correct associations.

Decreasing Δ from 0.6 to 0.4 results in a large boost to the average W2D precision (increasing between 6-11%), but also a large hit to the average W2D recall (decreasing between 4-5%). This change can be explained as a smaller magnitude in boosting during the handling of W2D possible associations. While the same association probabiliies are being boosted, many of them are no longer being boosted enough to meet the confidence threshold. Therefore only those with multiple supporting data points will have a probability to meet the confidence threshold, while those that only have one or a couple supporting data points, even if they are correct, will not meet the necessary confidence threshold, and will not be concluded as associations by the attack model.

While the absolute increase in precision is greater than the absolute decrease in recall in terms of percentage, relative to their existing values, the recall decrease is greater than the precision increase. That is, precision was already fairly high, around 80%, while recall was fairly low, around 21%, meaning an equal absolute change to both will affect recall relatively more. For example, using the largest precision increase where the WiFi penetration rate is equal to 80%, the average precision increase from 78.7% to 89.9% when decreasing Δ, meaning that the average precision saw an absolute increase of 11.2% and a relative increase of 14% ($\frac{0.899}{0.787}$). Using the same WiFi penetration rate, the average recall decreases from 21.8% to 17.6%. meaning that the average rcall saw an absolute decrease of 4.2%, but a relative decrease of

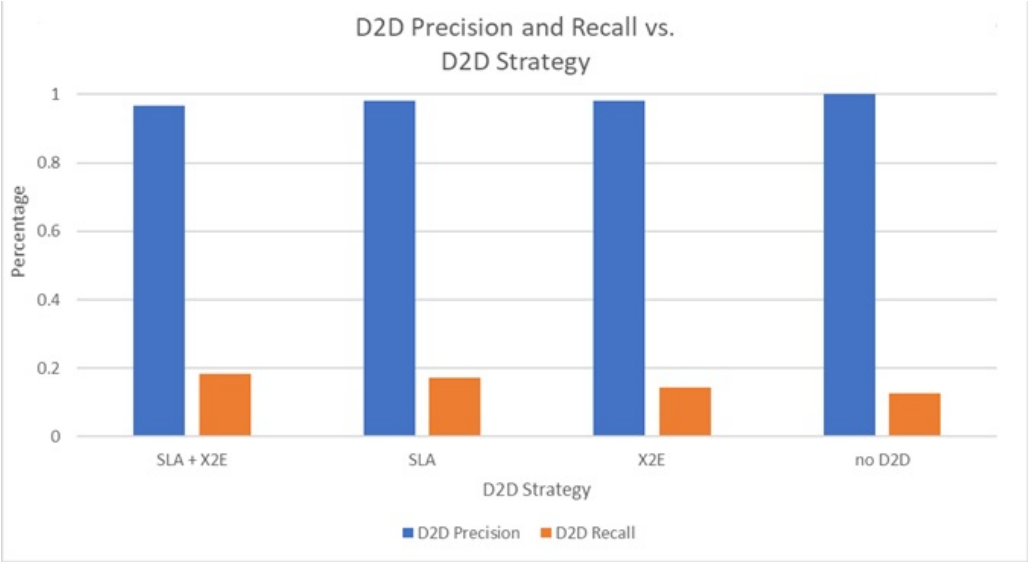


Figure 5.9: D2D Precision & Recall vs. D2D Strategy

19% ($1 - \frac{0.176}{0.218}$). Therefore, the relative change in average recall is greater than that of the average precision, even though the absolute change in the average precision is greater than that of the average recall.

5.4 D2D Techniques

In this section, I run my attack model using different configurations for the D2D matrix construction. I run the tests using all parts of the D2D matrix construction, just the Same Listener Attack, just the X2E Attack, and once without any D2D information. Note that the transitive property is applied to all configurations at the end. These tests were run on the Eichstätt simulation, using alternating WiFi penetration rates from 50-100% in 10% increments, and Δ values of $\Delta = 0.6$, $\Delta' = 0.9$, and $\Delta'' = 0.45$.

I begin by looking at the D2D precision and recall using different D2D strategies; Figure 5.9 shows my results. Note that even without any D2D matrix construction strategies, I still include the self-associations, where pseudonyms are associated to

themselves. This alone gives me a recall of 12.6%, which serves as a baseline for comparison. Using just the X2E Attack, I get a precision of 98% and a recall of 14.4%, meaning an actual gain of 1.8%. Using just the Same Listener Attack, I also get a precision of 98%, but a recall of 17.1%, meaning an actual gain of 4.5%. Therefore, while the X2E attack has the same precision as the Same Listener Attacker, it has a lower recall. For the X2E attack, I am able to have such a high precision through the use of the maximum weight to filter out the less likely to be correct edges when solving the biparitite graph. Without the filter, I would have a large number of false positives, which would, in conjunction with the cascading effects of the transitive property application at the end, lead to a much worse precision, with very little, if any, increase yield in recall. I do not use the possible associations in the D2D matrix to calculate the D2D precision or recall.

Using both the Same Listener Attack and X2E attack gave me a D2D precision of 96.7% and a recall of 18.4%. The slight dip in precision is most likely due to the combined false positives from both the Same Listener Attack and X2E Attack, as both did not have 100% precision. I also looked into applying the transitive property after the Same Listener Attack, not just at the end. However, this strategy gave me slightly poorer performance, most likely due to any false positives in the Same Listener Attack that may cause further cascading false positives. Therefore I opted to simply apply the transitive property at the end. I also chose to run the X2E Attack after the Same Listener Attack because the Same Listener Attack is a standalone attack, while my X2E Attack had been modified from the original attack [6] to take advantage of any previously found D2D associations.

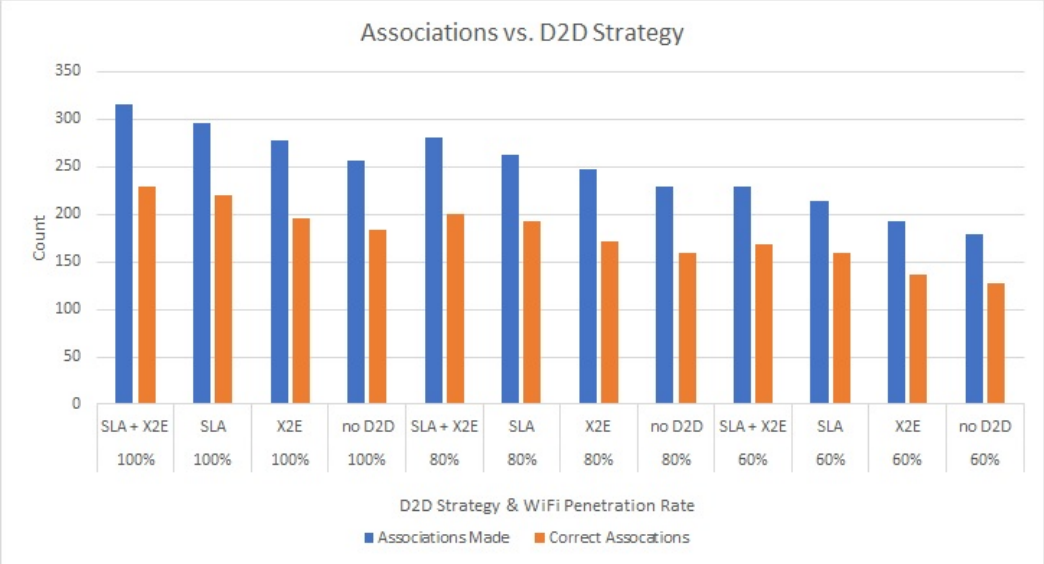


Figure 5.10: Associations vs. D2D Strategy

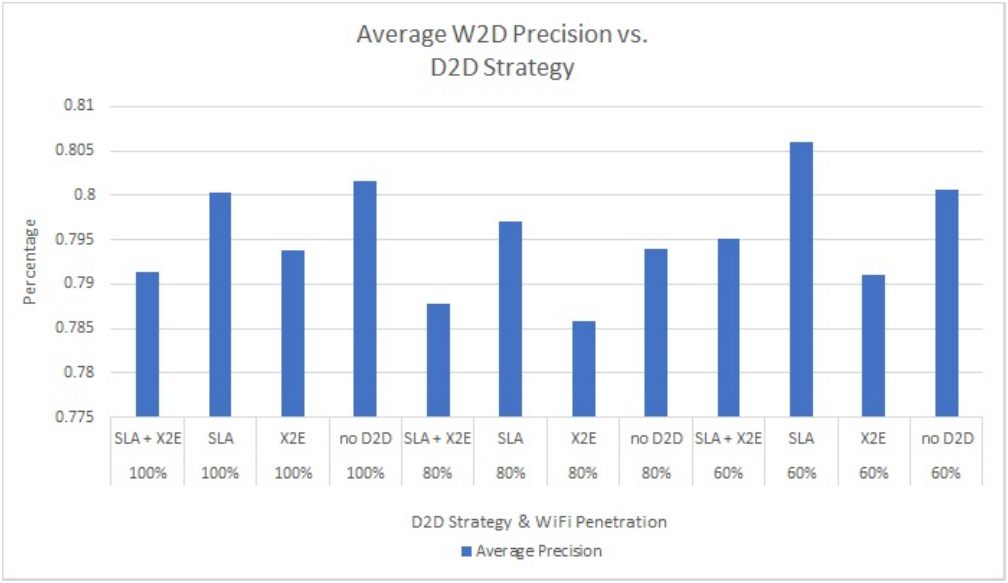


Figure 5.11: Average W2D Precision vs. D2D Strategy

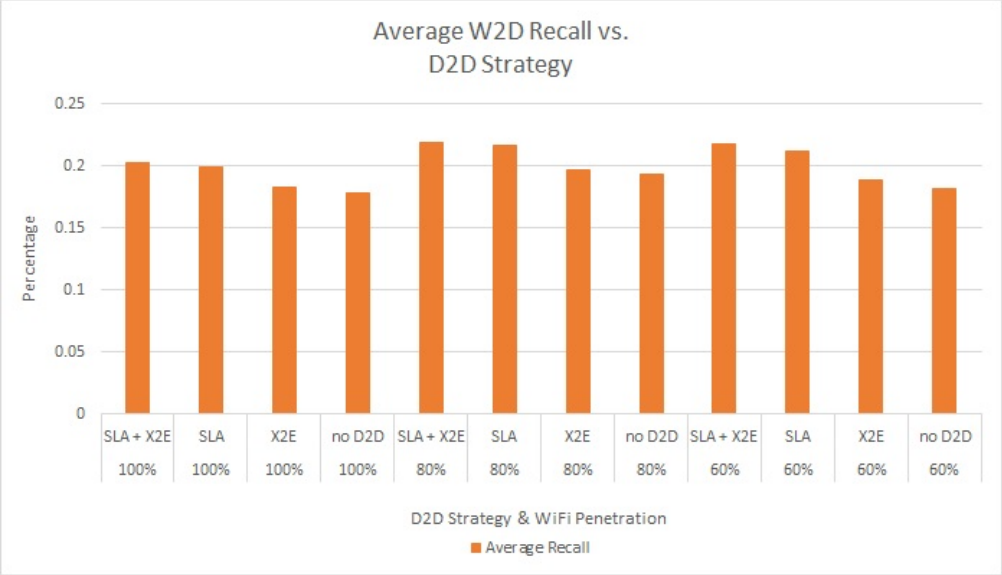


Figure 5.12: Average W2D Recall vs. D2D Strategy

The results of my experiments monitoring how different D2D matrix construction strategies affect W2D assciations are shown in Figure 5.11, Figure 5.12, and Figure 5.10. The average W2D precision is consistent across all configurations, ranging between around 79-80%. For the average W2D recall, using both D2D attacks gave me the best results, followed by the Same Listener Attack, the X2E Attack, and then no D2D association information. This is similar to the D2D association recall performance comparisons. Because precision stays the same, but recall drops as I use less effective D2D matrix construction strategies this means that I am making more guesses as I employ more effective D2D strategies. The new guesses are coming from my boosting. For example, given that the probability of w_1 and d_1 being associated were to be boosted to the point of meeting the confidence threshold, and d_1 and d_2 were previously found to be associated, then if there was no D2D association information or if I did not effectively use the information, then only d_1 would be found to be associated to w_1, while d_2 is actually associated to w_1 as well, but their association probability was not boosted accordingly. This is why I employ the use of the

Level 2 boosting, to use the D2D association information to expand the number of associations I can make, even if a small percentage of these are incorrect.

5.5 Weight Calculation

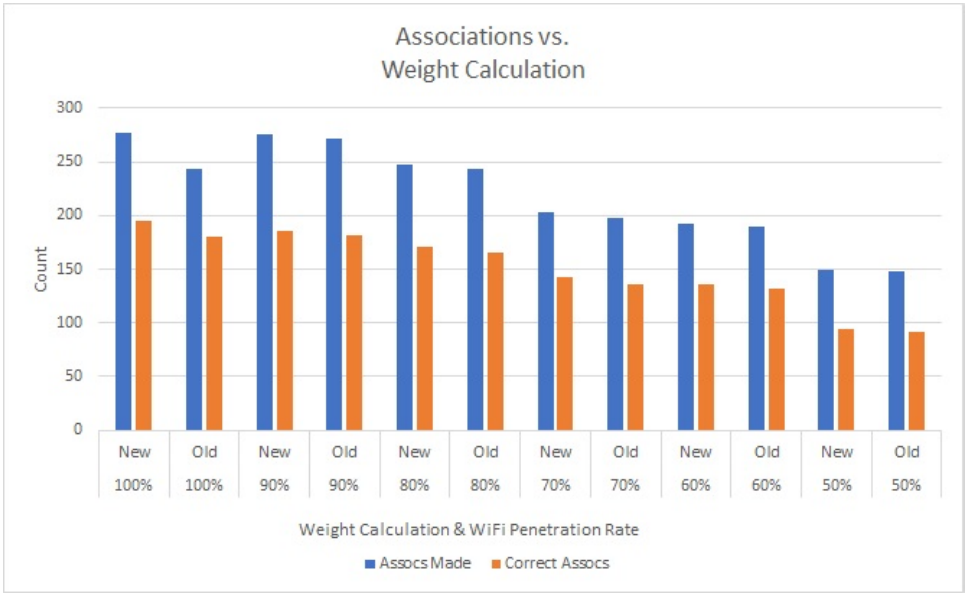


Figure 5.13: Associations vs. Weight Calculation

In this section, I detail the comparison between the previous methodology found in [6] and my methdology for weight calculations when constructing the bipartite graph in the X2E Attack. The previous methdology places less of an emphasis on the average travel time of vehicles than my methdology does, opting to make its significance relative to the actual travel time of a vehicle being observed. My methdology gives greater credence to the average travel time, while still penalizing actual travel times that greatly differ from the average. For these tests, I use alternating WiFi penetration rates from 50-100% in 10% increments, all steps of the D2D matrix construction only changing the weight calculations accordingly, and Δ values of $\Delta = 0.6$, $\Delta' = 0.9$, and $\Delta'' = 0.45$.

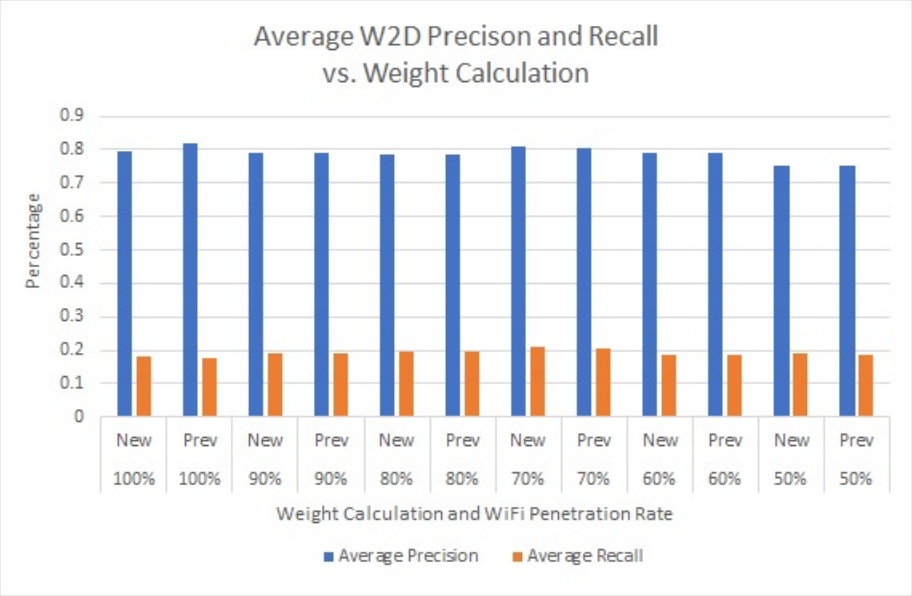


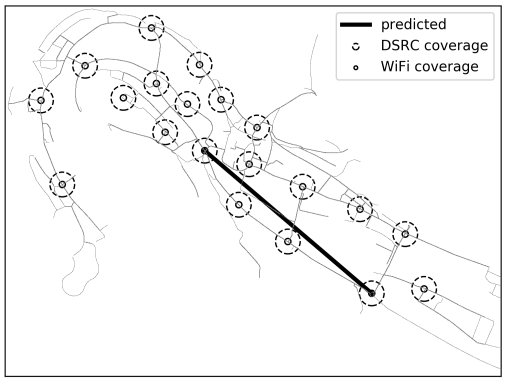
Figure 5.14: Average W2D Precision & Recall vs. Weight Calculation

For D2D association results, both results saw a precision of about 98%, but the previous method saw a recall of 13.8%, while the new method saw a recall of 14.4%. Similarly for W2D association results, as shown in Figure 5.13 and Figure 5.14, I see similar average precisions, with a small increase in the average recall. So while only a fairly small improvement in both D2D and W2D recall, my new weight calculation method consistently outperforms the previous method.

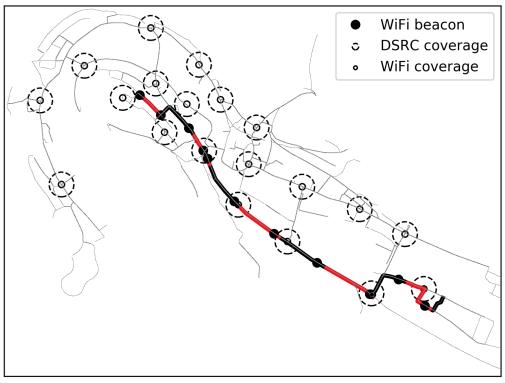
I correlate the more informative weight values to the information that I consider for the weight calculations. The previous method made the difference between actual travel time and average travel time a large factor in the weight calculations. Contrarily, my method places a heavier emphasis on the average travel time of vehicles, while still factoring in the difference between actual travel time and average travel time, albeit not to the degree of the previous method. Both methods place a large emphasis on the number of vehicles that previously traveled between two points.

5.6 Path Reconstruction

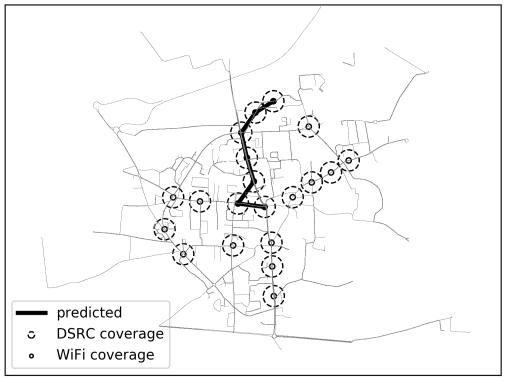
Finally, I reconstruct vehicles' paths using the technique explained in Chapter 4, then visualize and compare them to the vehicles' actual paths. Figure 5.15 shows three example comparisons between the reconstructed paths and actual paths, one for each map. The actual path is plotted with alternating colors (red and black) where a color change indicates a pseudonym change. I include these path reconstruction examples to illustrate the overall goal and usage of the associations made using the attack model.



(a) Reconstructed Path Eichstätt

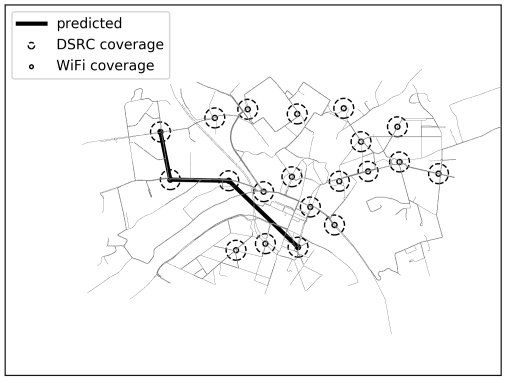


(b) Actual Path Eichstätt

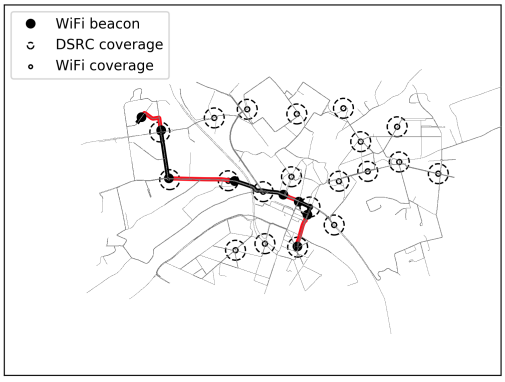


(c) Reconstructed Path Mions

(d) Actual Path Mions



(e) Reconstructed Path Melun



(f) Actual Path Melun

Figure 5.15: Reconstructed Path vs. Actual Path

Chapter 6

FUTURE WORKS

This chapter discusses possible future works that could expand upon the effectiveness of this attack on vehicular privacy in VANETs.

6.1 Expanding to k Signals

Our current attack model uses two generic signal protocols: long-distance pseudonym changing and short-distance non-changing. A future work is to expand the attack to take any arbitrary k number of signal protocols. This will allow for an even more generalizable attack model, one that works on any scenario involving radio signals.

6.2 Δ Tuning

Because our current Δ values were chosen through experimentation and trial and error, they are not necessarily the optimal values. Future work would include tuning the values using machine learning techniques to see if there are values that result in optimal effectiveness across many different maps. If there are no universal best values, then finding how factors such as map size or number of vehicles and pseudonyms affect the effectiveness of the Δ values.

6.3 Confidence Threshold and Second Most Likely Candidate

Another area of future research related to Δ tuning is the confidence threshold. Because adjusting Δ affects the amount of boost elements in the matrix get during matrix construction, adjusting the confidence threshold will also affect how much

boost is required to actually be considered an association. Furthermore, if for a given pseudonym the identifier it is concluded to be associated with is incorrect, then is the second most likely candidate in terms of probability the correct association? Future research can look into statistical analysis of incorporating secondary association into path reconstruction in the case of incorrect primary associations.

6.4 Distinguishing Vehicles Without WiFi Capability

Because our attack leverages WiFi identifiers to uniquely distinguish the number of vehicles in the area of observation, whenever the WiFi penetration rate is below 100%, we do not correctly identify the number of vehicles and have no means of discovering the number of vehicles without WiFi capability. We previously tried to solve this problem to an extent through our WiFi Null Boosting method, but due to the nature of the signal protocols heart rates, too many DSRC pseudonyms were boosted in their association to WiFi Null, thus yielding too many false positives to make the method useful. Furthermore, this method only tries to address the issue of which pseudonyms originated from vehicles without WiFi capability, not actually how many vehicles without WiFi capability there are. Future work would be to continue research into methods to distinguish the number of vehicles there are in the area of observation when not all vehicles share a common unique identifier.

6.5 Optimal Listener Placement

Our current attack model simply uses hand placed listener placement configuration for its listener placements. However, the attack could greatly benefit from a more optimal listener placement configuration. A previous Cal Poly Senior Project uses a genetic algorithm to find the optimal placement for listeners. A genetic algorithm is an algorithm that begins with a population of candidate solutions, in our case

the listener placement locations, and iteratively adjusts or mutates the population to optimize the solution to a problem. The effectiveness of the population is determined by a fitness function. By using the students algorithm, we can use our attack as the fitness function, with the average precision and average recall of WiFi-to-DSRC associations, and precision and recall of DSRC-to-DSRC associations be quantitative measurements of fitness.

Chapter 7

CONCLUSION

Maintaining vehicle security when expanding the technological applications of VANETs is a key component in preserving user privacy and safety. An important step in defending against vehicular privacy attacks is to know what attackers are capable of accomplishing. This thesis presents the attack model that I have developed and its effectiveness at associating pseudonyms.

Using previous research into pseudonym change strategies and VANET attacks, I have developed a passive attack mode that uses a general long-distance pseudonym changing protocol and a general short-distance non-changing protocol to uniquely identify vehicles and associate pseudonyms to their respective vehicles. This information can then ultimately be used to reconstruct a vehicle's path, allowing an attacker to track a numer of vehicles for an extended period of time. My attack model incorporates previously researched attacks into my new techniques, as well as implementing a more effective weight calculation in the commonly used Exit-to-Enter Attack.

Testing using different WiFi market penetration rates, that is the percentage of vehicles in the system that have WiFi capability, and different maps, my attack model consistently maintains a WiFi-to-DSRC association average precision of around 80% and an average recall of 23%, results that can be possibly even further improved by tuning different parameters in the attack model through the use of machine learning techniques. These results are comparatively better than previous association techniques.

Overall, this thesis helps outline the capabilities of a malicious attacker. The attack model can be used in future research to develop effective defense strategies to combat attackers and preserve user privacy and safety in VANETs.

BIBLIOGRAPHY

[1] Cal Poly Github. `http://www.github.com/CalPoly`.

[2] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz. Sumo–simulation of urban mobility: an overview. In *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*. ThinkMind, 2011.

[3] A. R. Beresford and F. Stajano. Mix zones: User privacy in location-aware services. In *Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on*, pages 127–131. IEEE, 2004.

[4] I. Bilogrevic, I. Aad, P. Ginzboorg, V. Niemi, J.-P. Hubaux, et al. Track me if you can: On the effectiveness of context-based identifier changes in deployed mobile networks. In *Proceedings of the 19th Annual Network & Distributed System Security Symposium (NDSS 2012)*, number EPFL-CONF-169827. Internet Society, 2012.

[5] D. Da Silva, T. Ann Kosa, S. Marsh, and K. El-Khatib. Examining privacy in vehicular ad-hoc networks. In *Proceedings of the second ACM international symposium on Design and analysis of intelligent vehicular networks and applications*, pages 105–110. ACM, 2012.

[6] D. Förster, F. Kargl, and H. Löhr. A framework for evaluating pseudonym strategies in vehicular ad-hoc networks. In *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, page 19. ACM, 2015.

[7] M. Gerlach and F. Guttler. Privacy in vanets using changing pseudonyms-ideal and real. In *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*, pages 2521–2525. IEEE, 2007.

[8] R. M. Ishtiaq Roufa, H. Mustafaa, S. O. Travis Taylora, W. Xua, M. Gruteserb, W. Trappeb, and I. Seskarb. Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study. In *19th USENIX Security Symposium, Washington DC*, pages 11–13, 2010.

[9] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[10] J. B. Kenney. Dedicated short-range communications (dsrc) standards in the united states. *Proceedings of the IEEE*, 99(7):1162–1182, 2011.

[11] M. Khodaei and P. Papadimitratos. The key to intelligent transportation: Identity and credential management in vehicular communication systems. *IEEE Vehicular Technology Magazine*, 10(4):63–69, 2015.

[12] J. Liao and J. Li. Effectively changing pseudonyms for privacy protection in vanets. In *Pervasive Systems, Algorithms, and Networks (ISPAN), 2009 10th International Symposium on*, pages 648–652. IEEE, 2009.

[13] A. Musa and J. Eriksson. Tracking unmodified smartphones using wi-fi monitors. In *Proceedings of the 10th ACM conference on embedded network sensor systems*, pages 281–294. ACM, 2012.

[14] H. Nissenbaum. Protecting privacy in an information age: The problem of privacy in public. *Law and philosophy*, 17(5):559–596, 1998.

[15] H. Nissenbaum. Privacy as contextual integrity. *Wash. L. Rev.*, 79:119, 2004.

[16] P. Papadimitratos, L. Buttyan, J.-P. Hubaux, F. Kargl, A. Kung, and M. Raya. Architecture for secure and private vehicular communications. In *Telecommunications, 2007. ITST'07. 7th International Conference on ITS*, pages 1–6. IEEE, 2007.

[17] J. Petit, F. Schaub, M. Feiri, and F. Kargl. Pseudonym schemes in vehicular networks: A survey. *IEEE communications surveys & tutorials*, 17(1):228–255, 2015.

[18] N. Plewtong. Modeling adversarial insider vehicle in mix zones. Master's thesis, California Polytechnic State University San Luis Obispo, San Luis Obispo CA 93407, 3 2018.

[19] M. Raya and J.-P. Hubaux. Securing vehicular ad hoc networks. *Journal of Computer Security*, 15(1):39–68, 2007.

[20] M. Raya, P. Papadimitratos, and J.-P. Hubaux. Securing vehicular communications. *IEEE wireless communications*, 13(5), 2006.

[21] K. Sampigethaya, M. Li, L. Huang, and R. Poovendran. Amoeba: Robust location privacy scheme for vanet. *IEEE Journal on Selected Areas in communications*, 25(8), 2007.

[22] M. L. Sichitiu and M. Kihl. Inter-vehicle communication systems: a survey. *IEEE Communications Surveys & Tutorials*, 10(2), 2008.

[23] J.-H. Song, V. W. Wong, and V. C. Leung. Wireless location privacy protection in vehicular ad-hoc networks. *Mobile Networks and Applications*, 15(1):160–171, 2010.

[24] B. Wiedersheim, Z. Ma, F. Kargl, and P. Papadimitratos. Privacy in inter-vehicular networks: Why simple pseudonym change is not enough. In

Wireless On-demand Network Systems and Services (WONS), 2010 Seventh International Conference on, pages 176–183. IEEE, 2010.

[25] M. Zimmer. Surveillance, privacy and the ethics of vehicle safety communication technologies. *Ethics and Information Technology*, 7(4):201–210, 2005.