DEGREE PROJECT IN INFORMATION TECHNOLOGY, SECOND LEVEL
STOCKHOLM, SWEDEN 2016

Spatial Replay Protection for Proximity Services

Security and privacy aspects

FREDRIK LINDBLOM

KTH ROYAL INSTITUTE OF TECHNOLOGY
INFORMATION AND COMMUNICATION TECHNOLOGY

Spatial Replay Protection for Proximity Services

Security and privacy aspects

Fredrik Lindblom

2016-08-09

Master's Thesis

Examiner
Gerald Q. Maguire Jr.

Academic adviser
Anders Västberg

Industrial supervisor
Noamen Ben Henda

Abstract

Proximity Services is a new feature in the 3rd Generation Partnership Project (3GPP) standard for mobile communication. This features gives the opportunity to provide services locally if the targets are sufficiently close. However, in the current version of the proposed specification, there is no protection against a malicious user tunneling messages to a remote location to give the impression of proximity.

This thesis proposes solutions to protect against such a spatial replay attack and evaluates these solutions based on how the user's integrity is preserved, their complexity, and the added overhead. It is not obvious today what the consequences of a spatial replay attack are and how serious such an attack could be. However, once the feature is deployed and people start using it, it could prove to be a major vulnerability.

The methods presented in this thesis could be used to prevent spatial replay in 3GPP or similar standards proximity services. The chosen method is a geographical packet leash based on a poly-cylindrical grid for which only a certain amount of Least Significant Bits of the grid cell identifier is included in the initial Discovery Message and the rest could be used in the calculation of the Message Authentication Code.

Keywords

Location Based Services, Proximity Services, Spatial Replay, 3GPP, Location Privacy

Sammanfattning

Proximity Services är en ny funktion inom 3rd Generation Partnership Project (3GPP) standard för mobil kommunikation. Den möjliggör att erbjuda tjänster lokalt om de tänkta användarna är tillräckligt nära. I den nuvarande versionen av specifikationen så finns det dock inget som hindrar en tredje part med onda avsikter från att tunnla meddelanden från den ursprungliga platsen till en annan som inte är i närheten för att ge intrycket till mottagaren att sändaren finns nära.

Det här examensarbetet föreslår lösningar för att begränsa nämnda attack och utvärderar dem efter hur de påverkar användarnas platssekretess, lösningens komplexitet och den overhead de innebär. Det är idag inte uppenbart på vilket sätt den nämnda attacken skulle kunna påverka användarna och hur allvarliga konsekvenserna kan bli, men när standarden är implementerad och eventuella användare tillkommer så skulle det kunna visa sig innebära en stor risk.

Lösningarna som presenteras i det här examensarbetet skulle kunna användas för att begränsa den här typen av attacker inom 3GPPs standard eller liknande baserade på närhet. Den metoden som har valts är ett ’geographical packet leash’ baserat på ett polycylindriskt rutnät för vilket endast en bestämd mängd minst signifikanta bitar är inkluderade i ett inledande Discovery Message medans resten kan användas i beräkningen av Message Authentication Code.

Nyckelord

Platsbaserade tjänster, Proximity Services, Spatial Replay, 3GPP, Platssekretess

Acknowledgments

I would like to thank my industrial supervisor, Noamen Ben Henda, for continuously supporting me through this thesis and giving me valuable advice. I would also like to thank Ericsson for providing me with the opportunity of doing this thesis.

Professor Gerald Q. Maguire Jr. has also been of great support and provided guidance which I am thankful for.

Stockholm, July 2016
Fredrik Lindblom

Table of contents

List of Figures

List of Tables

List of acronyms and abbreviations

3GPP	3rd Generation Partnership Project
ALUID	Application Layer User ID
AP	Access Point
A-GPS	Assisted-Global Positioning System
BLE	Bluetooth Low Energy
DNS	Domain Name System
DUCK	Discovery User Confidentiality Key
DUSK	Discovery User Scrambling Key
ECGI	E-UTRAN Cell Global Identifier
EPC	Evolved Packet Core
EPUID	EPC ProSe User ID
ETSI	European Telecommunications Standards Institute
FCC	(United States) Federal Communications Commission
GAD	Universal Geographical Area Description
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HPLMN	Home Public Land Mobile Network
HSS	Home Subscriber Server
HTM	Hierarchical Triangular Mesh
IERS	International Earth Rotation and Reference Systems
LBS	Location Based Services
LPPM	Location Privacy Protection Mechanism
MAC	Message Authentication Code
MCC	Mobile Country Code
MIC	Message Integrity Code
MIMO	Multiple Input-Multiple Output
MME	Mobility Management Entity
MNC	Mobile Network Code
OTDOA	Observed Time Difference Of Arrival
PDUID	ProSe Discovery UE ID
PFID	ProSe Function ID
PLMN	Public Land Mobile Network
ProSe	Proximity Services
RFPM	Radio Frequency Pattern Matching
RPAUID	Restricted ProSe Application User ID
SLP	SUPL Location Platform
SUPL	Secure User Plane Location
TETRA	Terrestrial Trunked Radio
UE	User Equipment
UMTS	Universal Mobile Telecommunications System
UTC	Coordinated Universal Time
U-TDOA	Uplink-Time Difference of Arrival
WLAN	Wireless Local Area Network
WLLID	WLAN Link Layer ID

1 Introduction

This chapter describes the specific problem that this thesis addresses, the context of the problem, the goals of this thesis project, and outlines the structure of the thesis.

1.1 Background

Devices capable of locating themselves are becoming something everyone has. In 2013, more than a billion smartphones were shipped and over 40% of the worlds mobile phones had support for one or more Global Navigation Satellite System (GNSS)[1]. As a result, services based on location are becoming more and more common. A user's location is considered by most people to be a part of their personal privacy, but still there are many applications (apps) that have user agreements which allow the app to share the user's location with third parties. However, the location data that can be retrieved by the network operator are protected by law in a number of countries (such as Sweden and Finland) and must not be disclosed without either a court order or the user's written permission, see Section 2.3.1.

A local form of Location Based Services (LBS) is the proposed 3GPP standard Proximity Services (ProSe) (see Section 2.2). Proximity Services enables User Equipment (UE) within a maximum range of 500 meters to find and, if desired and close enough, to communicate directly with each other. This feature does not depend on UEs utilizing the same network operator, being connected to the same cell, or for Public Safety UEs to even be within any network's coverage. Each UE broadcasts its presence and the type of services it offers, hence when a UE hears such a broadcast these UEs can communicate directly with each other.

1.2 Problem definition

In the current version of the proposed ProSe standard, the only limitation on how far apart UEs can be is the 16 second time period during which the ProSe messages are valid and that the messages can reach the other UE. Normally, each UE's range would be limited to roughly 500 meters due to the maximum allowed emitted signal strength. However, as shown in Figure 1-1 an attacker UE_1 could tunnel messages from UE_A over the Internet to UE_2, tricking UE_B into believing that UE_A is nearby. This is referred to as a spatial replay attack.

The exact consequences of such an attack are hard to foresee, but the incorrect assumption by the services of the UE's being in proximity could pose a threat to some services in the future or simply result in inconvenient situations for the users.

As with LBS in general, there are also issues concerning the user's privacy. There are multiple Location Privacy Protection Mechanisms today that could be implemented (see Section 2.3.4). However, LBS has only recently begun to be widely deployed, therefore some of the privacy protection mechanisms might not be applicable, while others are infeasible. Since a LBS is based on the location of the UE, increased location privacy could reduce the quality of the service or even prevent it from working at all. While at the same time, insufficient location privacy might lead to users not utilizing any forms of LBS.

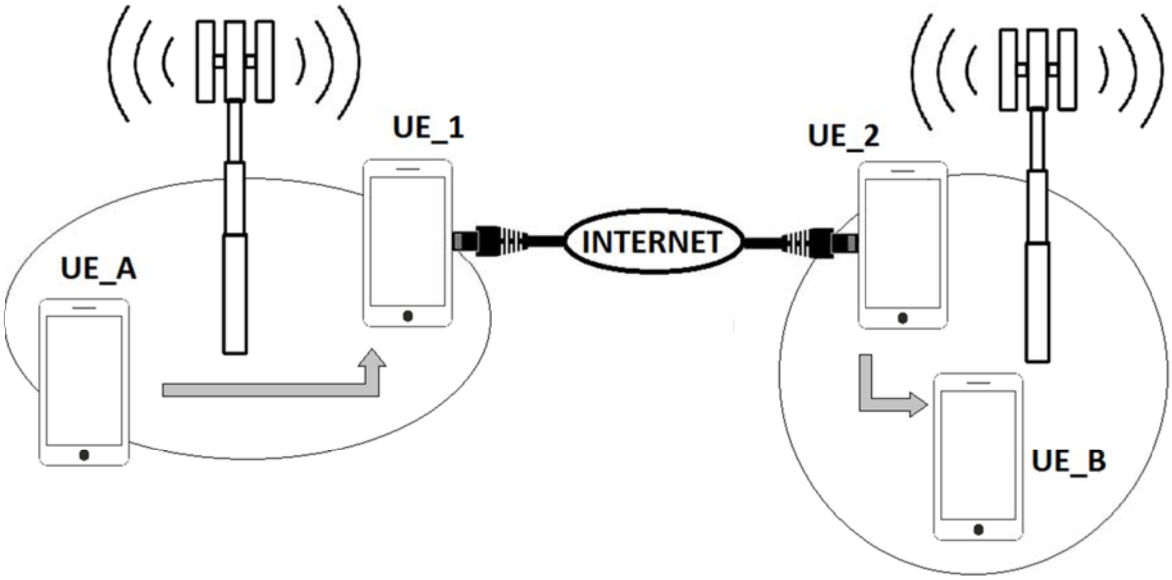

Figure 1-1: Spatial replay by 3rd party

1.3 Purpose

The results from this thesis could be implemented in the standard for Proximity Services by 3GPP and other standards that use location for a similar purpose, as systems using these standards could be vulnerable to replay attacks. An example of such a standard is the 3GPP-LTE based system for Vehicle-to-Everything (V2X) communication [2].

The proposed solutions should protect the user's integrity by not revealing their exact location while using ProSe or it might minimize the exposure of the user's location.

1.4 Goals

The primary goals of this thesis project are:

1. Present existing solutions to prevent replay attacks,
2. If necessary design one or more new solutions to prevent replay attacks, and
3. Evaluate the all solutions based on provided security, location privacy, added overhead, and complexity.

1.5 Research Methodology

This thesis has adopted the design science research method, as the focus is on the evaluation of *potential* solutions for preventing replay attacks according to a stated set of metrics (as per the third goal above).

1.6 Delimitations

This thesis will **not** include an actual implementation, but rather only describe potential solutions that could be implemented. The evaluation of these methods will mainly be limited to the security and personal integrity aspects related to location and spatial replay. Moreover, the thesis will focus only on an evaluation of potential solutions in the context of ProSe. Therefore, the goal is **not** to develop generic protection methods (although the proposed solutions may be applicable to other standards), but rather to develop solutions for ProSe (or identify aspects of ProSe that need modifications).

1.7 Structure of the thesis

Chapter 2 presents relevant background information and related work. Chapter 3 describes the methodology to be used in this thesis project. Chapter 4 evaluates some current vulnerabilities and existing solutions. Chapter 5 proposes some new solutions. Chapter 6 presents the analysis of the solutions presented in Chapter 5. The thesis concludes with some conclusions, suggestions for future work, and some reflections about the thesis in a larger context.

2 Background

This chapter presents the background necessary for the reader to understand what a location based service is (Section 2.1) and what a proximity service is (Section 2.2) and some of the problems that can arise concerning location privacy (Section 2.3). Mechanisms to protect the user's location privacy are presented in Section 2.3.4. Section 2.4 describes the concept of spatial replay. Section 2.5 presents a survey of related work. Section 2.6 gives a summary of this chapter.

2.1 Location Based Services

There is no common definition of what a Location Based Service (LBS) is. However, the definition most relevant to this thesis is given by 3GPP in [3], where an LBS is defined as a "service provided either by teleoperator or a 3rd party service provider that utilizes the available location information of the terminal" [3]. Additionally, there are other definitions, such as that given by Jochen Schiller and Agnès Voisard: "services that integrate a mobile device's location or position with other information so as to provide added value to the user" [4]. In most sources LBS is used interchangeably with *location service*, while 3GPP distinguishes between a *location service* and a *location **based** service*. 3GPP uses the term location services to refer to "a network provided enabling technology consisting of standardized service capabilities which enable the provision of location based applications" [5]. In other words, 3GPP defines location services as ways to localize a target in order to be able to provide location data to a 3rd party; for example, to emergency services to aid with emergency calls. In this thesis the term "*location services*" will **not** be used to avoid ambiguity.

Fundamental to any LBS is that it must be possible to locate the device to which or for which a location based service is to be provided. Methods for providing this location as input to the LBS are described in Section 2.1.2 along with some information about their accuracy. However, before presenting the details of the operation of a LBS, Section 2.1.1 describes how a LBS might be used. Finally, Section 2.1.3 describes how locations can be encoded.

2.1.1 Use cases

LBS can be used for many different purposes and a lot of purposes are probably yet to come. In [6] Axel Küpper gives a thorough introduction to LBS, and describes many use cases for them. He classifies LBS as either reactive or proactive, with the difference between them being that a reactive services has to be explicitly activated by the user while a proactive is activated by certain conditions (such as being in a specific location).

An example of a reactive LBS would be a service returning nearby points of interest, such as a restaurant or an automatic teller machine. According to Axel Küpper this is the most widespread LBS so far. Another use case could be a parent activating a tracker on their child's mobile phone when they need to locate their child.

An LBS giving information about nearby points of interest could also be made proactive. An example of such a proactive LBS would be one that waits for the user, e.g. a tourist, to enter a new area and give information to this user about what is available in this new area. Another proactive LBS could be made for car travelers to provide updates about the traffic conditions of the highway ahead of them or of a roadway that they are approaching. The LBS could alert the user if they are approaching a traffic jam or an area with construction work.

2.1.2 Locating methods

There are many ways to obtain a location as input to a LBS. The most obvious method is the Global Positioning System (GPS) and other global navigation satellite systems (GNSSs). These methods require that the UE has a suitable receiver to receive signals from multiple satellites, but there are also methods based on hardware already present in many mobile phones (such as receivers for other purposes, e.g. Wi-Fi).

Implementation of locating methods in cellular phones was heavily driven by the United States Federal Communications Commission's (FCC) decisions to require operators to be able to provide to a public safety answering point the location of a phone making an emergency call (see Enhanced 911 Emergency Calling Systems (in FCC 99-27 [7], FCC 96-52 [8], and others)).

2.1.2.1 *GNSS / A-GPS*

In addition to the United State Government's GPS system, there are other GNSSs, such as Russia's "Globalnaya Navigazionnaya Sputnikovaya Sistema" (GLONASS), Europe's Galileo system, and China's Beidou. Currently, only GPS and GLONASS are fully operational globally. Since these systems rely on satellite signal reception, all of the systems generally suffer from lack of signal strength indoors and radio shadows caused by buildings and vegetation outdoors.

The time required to achieve a GPS lock* with a satellite depend on when the receiver last had a lock. If it has been more than 30 seconds since the last lock, then the receiver has to start over, while if less the lock can quickly be regained. If the last lock was less than a second ago, the lock will not need renewal. To achieve a GPS lock is more time and energy consuming than the effort to just decode the data needed to calculate a position [9]. Additionally, the receiver needs to achieve a lock with at least 4 satellites to accurately determine x, y, z, and time.

Aaron Carroll and Gernot Heiser measured the GPS module's power consumption in an Openmoko Neo Freerunner to be around 143 mW. Other measurements of mobile phone GPS power consumption have been made, e.g. 370 mW for a Nokia N95 [10] and 230 mW for a HTC Dream [11].

To reduce the time to lock on to satellites during a cold start and to provide better location data in poor satellite signal conditions Assisted-GPS (A-GPS) can be utilized. A-GPS can supply orbital data of the satellites and also precise time, if needed by the receiver, from the cellular network to reduce the time needed for decoding. Remote computing resources can also assist in processing the received signals to both provide a more accurate location and reduce the load on the UE. It has also been investigated to offload the entire calculations to the cloud. In [9] an energy consumption 3 orders of magnitude less was achieved for a single location update.

2.1.2.2 *CELL ID / Media Access and Control address*

One of the simplest methods of determining a phone's location is to use the CELL ID of the nearest cellular base station or use the CELL IDs of several nearby base stations. The information has to be complemented with Public Land Mobile Network (PLMN) ID and possibly other information depending on the mobile communication standard used. This information can be combined with the geospatial location of these cellular base stations to estimate the phone's position. The same approach can be used with Media Access and Control addresses of Wireless Local Area Network (WLAN) access points if their location is known.

* This requires both synchronization with the pseudo-random sequence and adjustment of the receiver's frequency (because to the Doppler shift due to the relative motion between the receiver and the satellite).

This can be done both locally by the UE or, rather than to store all necessary data, by a third party service. Google offers a service in which the unique IDs (CELLID or Media Access and Control address) are sent to them and looked up in databases. If they have location information for these IDs, then the user's location is estimated and returned. In addition to CELLIDs, one should also supply the type of radio used (if available). It is also possible to send signal strength and time needed for a signal to reach the base station for GSM CELLIDs, but this information is currently not used by Google [12]. Moreover, when data is shared with a third party the result is reduced location privacy. In contrast, Intel's Privacy Observant Location System (POLS) is an early implementation that does not require sending data to a third party [13] and free databases exist, such as OpenCellID [14].

2.1.2.3 *Network time of arrival based*

Uplink-Time Difference of Arrival (U-TDOA) is based on a known signal sent from the UE that is received by at least four network nodes. The advantage of this method is that it does not require the UE and the network nodes to be synchronized in time, hence the actual time the signal is sent can be unknown. This freedom from synchronization of clocks is due to the fact that this method only requires the *time difference of arrival* to the different nodes when calculating the UE's location. The time difference of arrival from any two of the nodes gives a hyperbola of locations of the UE from which the signal *could* have originated. By creating considering pairs of different nodes the UE's location is computed based upon the intersection of three hyperbolas. This method does not require any special hardware in the UE other than its ability to communicate with the network [15].

Observed Time Difference Of Arrival (OTDOA) is based on the same principle, but in this case the location is calculated by the UE, rather than the network. There are many variants of this method to address various practical issues, such as being too close to one of the network node [16].

2.1.3 Location Coding

Location identifiers can be based on different references. GPS is based on World Geodetic System (WGS) 84. The origin of the coordinate system is at the earth's center of mass and the definition of zero longitude is the International Earth Rotation and Reference Systems (IERS) Reference Meridian (nearly the same as the Greenwich meridian). Latitude is defined as the angle between the equatorial plane and a radius from the earth's midpoint to its surface, while longitude is defined as the angle from the radius to the reference meridian.

3GPP uses WGS 84 as its reference system for location coding. In 3GPP document TS 23.032 [17] different Universal Geographical Area Description (GAD) shapes are defined: ellipsoid point, polygon, and ellipsoid arc. An ellipsoid point is a point on the surface of an ellipsoid. It is defined by a latitude and a longitude. A point can also have an altitude that is the distance over the nominal sea level. It can also have an uncertainty circle or ellipsoid. The circle or ellipsoid defines points within the range that are part of the uncertainty of the location.

To define shape type, 1 byte is used and then latitude and longitude is encoded as 3 bytes each. 1 byte is used to encode an uncertainty circle and 4 bytes to encode an uncertainty ellipsoid.

The latitude coding is given in Equation 2-1 and the longitude coding in Equation 2-2.

$$N \leq \frac{2^{24}}{360}\mathrm{X} < N+1 \qquad \text{2-1}$$

$$N \leq \frac{2^{23}}{90} X < N + 1 \qquad \text{2-2}$$

N is the coded number and X the latitude/longitude it encodes. For the latitude, when N=2^{23}-1, the range also includes N+1. The latitude is coded with 24 bits of which 1 is a sign bit and the longitude is coded in 2's complement in 24 bits.

The standard also provides a means of encoding velocities and bearings, but does not define encodings for acceleration.

2.2 Proximity services

One of the major driving forces for the introduction of proximity services is the desire to merge the currently separate commercial cellular networks and the (dedicated) public safety networks (such as Terrestrial Trunked Radio (TETRA) and P25). This is driven both due to the cost of maintaining dedicated public safety networks and the realization that the public safety networks have fallen far behind the capabilities of the commercial cellular networks. As a result, commercial subscribers with camera equipped smartphones have broadband streaming of multimedia, while public safety systems offer at most hundreds of kilobits per second greatly hampering public safety officials.

Proximity services offer two features that are important for public safety activities: (1) discovery of and direct communication with nearby UEs and (2) group calls. Today proximity services (ProSe) is currently in the process of being standardized by 3GPP. ProSe is sometimes referred to as Proximity-based Services and different terms even coexist within the same 3GPP work group.

Figure 2-1 shows the ProSe architecture as presented in [18]. In this figure, both UE A and UE B are subscribed to the same PLMN and neither UE is roaming[*]. When the UEs are subscribed to different PLMNs, then another interface called PC6 is added between the ProSe functions in the different PLMNs. Each PLMN has its own instance of everything, i.e., Home Subscriber Server (HSS) and Secure User Plane Location (SUPL) & Location Platform (SLP). Details of this architecture are given in the next subsection.

The other driving force for ProSe is Qualcomm's LTE Direct [19] which has been incorporated as a part of ProSe. This technology allows for a UE to find other nearby UEs. As a result, the ProSe specification includes many different ways to find nearby UEs to interact with, multiple methods of communication once a connection has been established, and extra features for Public Safety use (and other specialized use cases). The goal of these different ways, methods, and features is to provide convenient features for the user, to reduce the load on the network, and to efficiently utilize the available frequency spectrum.

[*] Roaming is not expected to affect the approach used in this thesis.

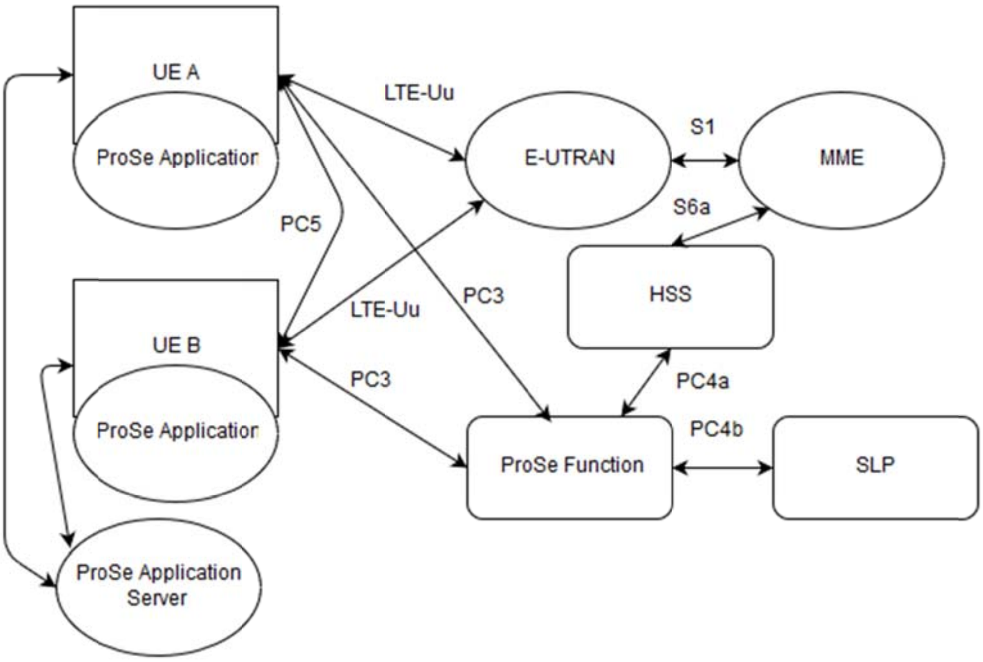

Figure 2-1: Architecture of ProSe

2.2.1 Architecture and interfaces

The central parts of the architecture relevant for this thesis are the ProSe Function, ProSe Application Server, and the ProSe Application on the UE. These elements and the others shown in Figure 2-1 are list in Table 2-1, while all of the ProSe reference points are listed in Table 2-2.

Table 2-1: Elements of ProSe architecture

Network node	**Description**
ProSe Application	The ProSe Application is the application running on the UE. This application uses the 3GPP API to use the features in ProSe.
ProSe Function	The ProSe Function handles the communication with non-ProSe specific parts of the network like with the HSS to provide authentication of the UEs requesting to use ProSe.
ProSe Application Server	The ProSe Application Server is responsible for storing lists of IDs (such as ProSe Discovery UE ID and Restricted ProSe Application User ID (RPAUID)) as well as metadata for applications. It also maintains permission information for restricted discovery.
HSS	The HSS is a central database of subscriber information. It handles authorization and authentication
SLP	Handles the user locations for Evolved Packet Core (EPC)-based Discovery.
MME	Receives subscription information from the HSS related to ProSe, maintains a list of Remote UEs connected to ProSe UE-to-Network Relay and forwards information to the SGW.
E-UTRAN	Evolved Universal Terrestrial Access Network is the access part of the Evolved Packet Core. It consists of eNodeBs that communicate directly with the UEs.

Table 2-2: ProSe Reference Points

Interface	Purpose
PC1	Reference point between the ProSe application running on the UE and the ProSe Application Server to define signaling requirements.
PC2	Reference point between the ProSe application and the ProSe Function. Used for defining interactions between them.
PC3	Reference point between the UE and the ProSe Function and used to define their interactions.
PC4a	Reference point between the HSS and the ProSe Function. Used for providing subscription information.
PC4b	Reference point between the SLP and and the ProSe Function. Used to handle the location of users in EPC-level ProSe Discovery.
PC5	This interface provides both the control and user plane for ProSe Direct Discovery, Direct Communication, and UE-to-Network Relay.
PC6	Used for communication by ProSe Functions in different PLMNs when not roaming. E.g. by ProSe Functions for EPC-level ProSe Discovery.
PC7	Reference point between the ProSe Function in VPLMN and in Home Public Land Mobile Network (HPLMN).
S6a	In ProSe, this interface is used to download subscription information to MME during E-UTRAN attach procedure and to inform the MME of subscription information changes in the HSS.

2.2.2 Use cases

In the 3GPP feasibility study [20] several use cases are given for every part of the standard. An example of Open Discovery occurs when the user is looking for a restaurant and walks within the proximity range of a restaurant utilizing the service. In this case the user will be notified about this restaurant's existence. When trying to find parking near the restaurant an application using ProSe could assist the user in finding a nearby parking spot and paying for parking.

An example of Restricted Discovery occurs when the user is trying to find a friend or colleague. To protect the user's privacy, this discovery should be limited to users who are actually friends or colleagues. This restriction is realized by requiring that such a discovery be permitted. Another use case occurs when two users have an active data session with each other via the network, but are in proximity of each other. In this case the session is moved to a ProSe communication path, thus reducing the communication delay and shifting the load off of the core network. This session can then be moved back to the core network when the UEs are no longer in proximity.

2.2.3 Identifiers and subscriptions

The permissions to use different features in ProSe are stored in the user profile subscription information in the HSS. The following permissions (sub-permissions have been omitted) are available to all UEs: ProSe Direct Discovery, EPC-level ProSe Discovery, and EPC-support WLAN Direct Discovery and Communication. There are additional permissions exclusive for public Safety users: ProSe Direct Communication, one-to-one and one-to-many, ProSe UE acting as

UE-to-Network Relay, and Remote UE access to UE-to-Network Relay. Permissions can be added and revoked any time and each permission can also be restricted to a specific PLMN.

There are many different types of identifiers (IDs), some of these are listed in Table 2-3.

Table 2-3: ProSe IDs and their purposes

Type of ID	Acronym	Purpose
Application ID		A unique identifier of an application.
Application Code		The code broadcast on PC-5 in Open Discovery.
ProSe Query Code		The code broadcast on PC-5 in Restricted Discovery Model B.
ProSe Response Code		The code used as a response when a ProSe Query Code matches the discovery filter in Restricted Discovery Model B.
ProSe Restricted Code		The code broadcast on PC-5 in Restricted Discovery Model A.
Restricted ProSe Application User ID	RPAUID	A RPAUID is mapped by the ProSe Application server to the application user identity to hide it from the 3GPP network

2.2.4 Discovery models

The method of discovering nearby UEs in ProSe is called ProSe Discovery and is described in [18] and [21]. There are two kinds of discovery (EPC-level and Direct Discovery) differing in whether any network node is used or not. In Direct Discovery the PC5 interface is used to send discovery messages for Open Discovery and Restricted Discovery. In EPC-level discovery there is no direct communication link between the UEs, instead the network handles the discovery.

2.2.4.1 *Open Discovery*

In Open Discovery two roles are defined: Announcing UE and Monitoring UE. Before using any of the ProSe Direct Discovery models, the UEs must first request permission to use the service. The sequence diagram of ProSe Open Discovery is shown in Figure 2-2.

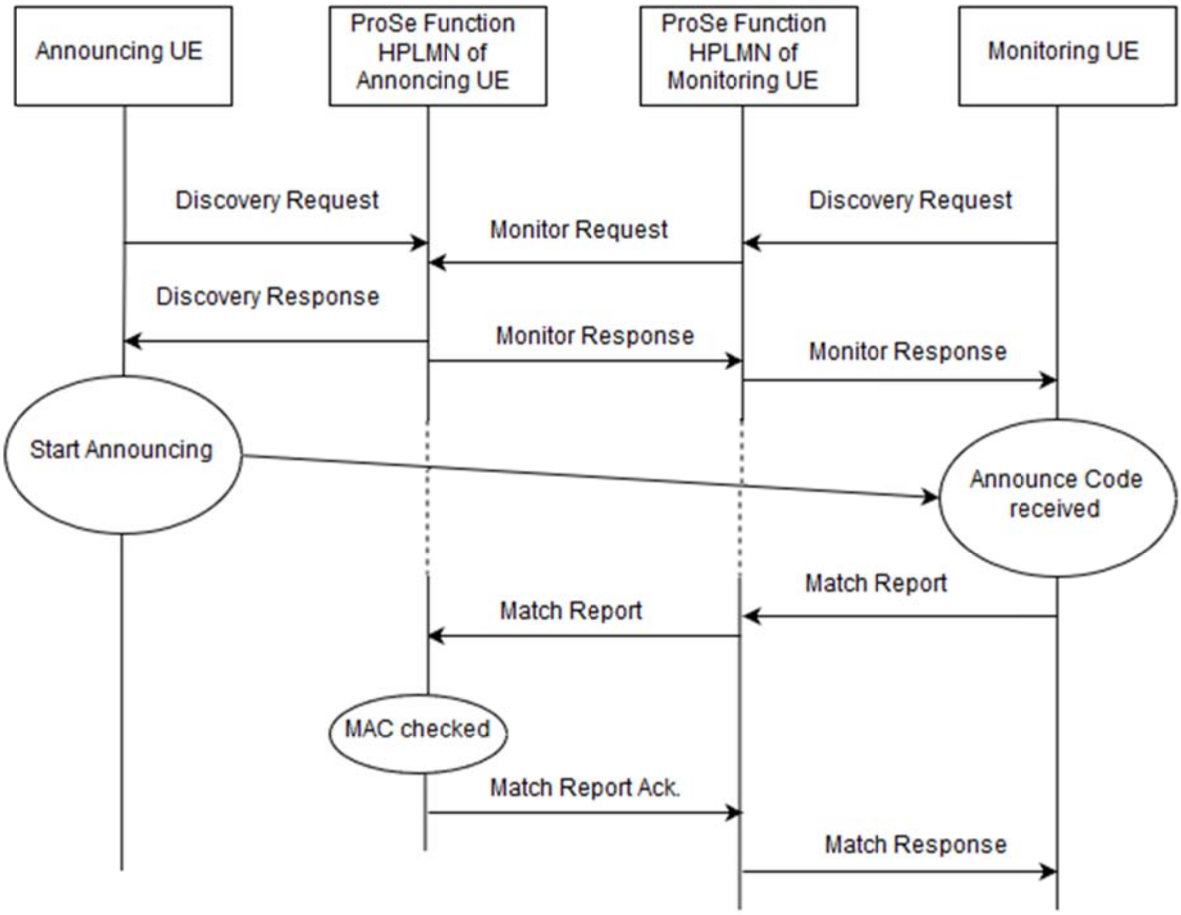

Figure 2-2: **Sequence diagram of ProSe Open Discovery**

In the announce request for Open Discovery, a ProSe Application ID, the UE identity, and the Discovery Entry ID are sent over the PC3 interface to the ProSe Function among with some additional data items (these are less relevant to this thesis). The ProSe Function queries the HSS for subscription information of the UE to determine if this subscriber is allowed to use the requested functionality and then checks if the UE is allowed to use the requested Application ID. If both checks are passed, then the ProSe Function returns a Discovery Response to the requesting UE with a Discovery Key, an Application Code, the current Coordinated Universal Time (UTC)-based time at the ProSe Function, and the max offset. The max offset is the maximum allowed difference between the UE clock and the UTC-based counter associated with the discovery slot. The UE can now start sending discovery messages containing the received Application Code over the PC5 interface.

The Monitoring UE sends a similar message to its ProSe function, but specifies that it wishes to monitor (look for) certain Application IDs. A check is done with the HSS to see if this UE has the relevant permission(s), but the PLMN ID is used to restrict the UE to its current PLMN. The ProSe Function that corresponds to the Application Code, if other than the current ProSe Function, is contacted to retrieve the Application Codes and an Application Mask for the Application ID. If the UE is authorized to monitor these Application IDs in this PLMN, then the ProSe function returns a Discovery Filter valid for a certain time corresponding to the Application Mask.

The discovery messages sent by the announcing UE over PC5 can now be monitored by the Monitoring UE and if it finds an Application Code matching the Discovery Filter, a Match Report is sent. The application mask allows both full and partial matches of Application codes. A Match Report contains the Application Code that matched the Discovery Filter, the UE Identity, the UTC-based time of the match as observed at the Monitoring UE, and the MAC from the discovery message.

The ProSe Function sends the MAC and the time value from the Monitoring UE to the HPLMN of the Announcing UE to get the MAC verified. If the MAC is valid, then the ProSe Function returns an acknowledgement to the Monitoring UE of the passed authentication check of the message, its current UTC-based time, a timer for when the match needs to be refreshed, and the Application ID.

The Monitoring UE now knows it is in ProSe range of an UE with the Application code it was looking for and can now proceed with whatever action it intends when in proximity of such a UE. These actions differ depending on whether the UE is a public safety UE or not (see Section 2.2.5.1).

2.2.4.2 *Restricted Discovery*

Restricted Discovery exist in two different models: Model A and B. Model A is when one of the UEs announces “I am here” and provides some information about itself. This is the model supported for Open Discovery. In Model B, a UE instead queries asking “Who is there?” or “Are you there?”. Model B is only supported for Restricted Discovery. The difference between Open Discovery and Restricted Discovery is that in Restricted Discovery permission is needed to discover the other UE.

2.2.4.2.1 Discovery Model A

Model A for Restricted Discovery is very similar to Open Discovery. The differences are that the Announcing UE, when it sends the request to announce, also includes its Restricted ProSe Application User ID (RPAUID) and when the ProSe function gives permission to announce, in addition to the data sent for Open Discovery, the ProSe Function also includes Code-Sending Security Parameters in the message. For the Monitoring UE, the same change applies; it receives Code-Receiving Security Parameters from the ProSe Function when given permission to monitor. What the parameters include is not really clear, but the necessary information for the security features in Section 2.2.6 should be among them.

When the Discovery Filter matches the Application Code of a discovery message, a request for permission to discover the other UE is sent to the ProSe Function. The ProSe Function checks with the HSS that this UE has permission to use Restricted Discovery and finds the target RPAUID for the Restricted Code received from the Monitoring UE. The Target RPAUID and the RPAUID of the Monitoring UE are sent to the ProSe Application Server corresponding to the Application ID and if this discovery is authorized, then the ProSe Application Server sends back the ProSe Discovery UE IDs (PDUIDs) of the two UEs. A discovery acknowledgement is sent to the monitoring UE and optionally the ProSe Function of the Announcing UE is also sent a discovery acknowledgement.

2.2.4.2.2 Discovery Model B

In Model B, the actors are not called Announcing UE and Monitoring UE anymore, but rather Discoverer and Discoveree. Figure 2-3 shows the sequence diagram of Restricted Discovery Model B.

The Discoveree starts with a Discovery Request similar to the one in Model A, but specifies the request type to be for Restricted Discovery, Discovery Model A, and that it wants to be the Discoveree. As in Model A, the request contains the UE’s RPAUID, UE identity, Application ID and Discovery Entry ID. As in the case for Discovery Model A, the ProSe Function checks with the HSS whether this UE is authorized to use the requested discovery model and whether access to the

Application corresponding to the Application ID is authorized. If the ProSe Function cannot verify the ownership of the supplied RPAUID and match it with its corresponding PDUID, then the ProSe Application for the specified Application ID is queried to verify this ownership and to receive the PDUID.

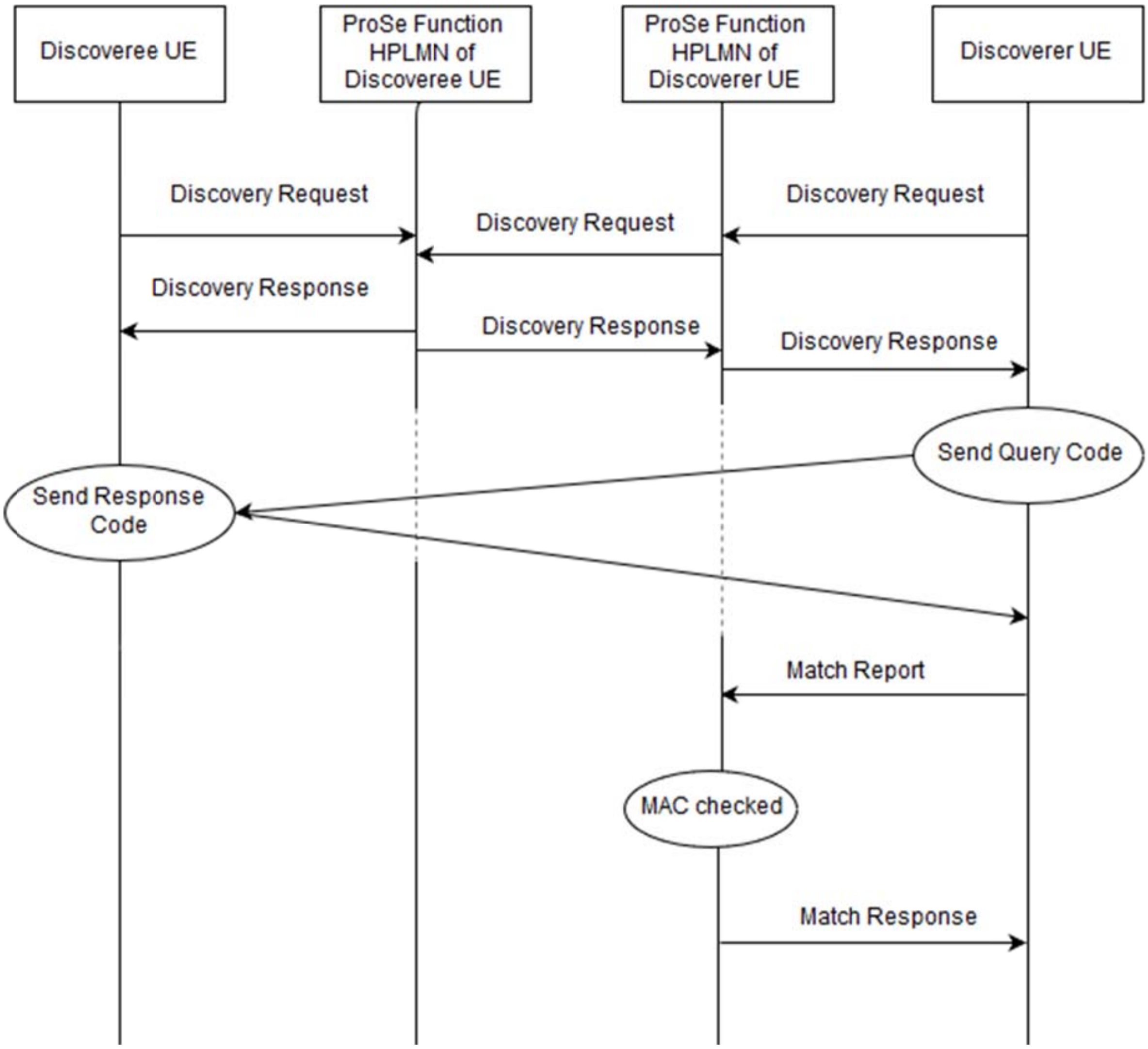

Figure 2-3: **Sequence diagram of Restricted Discovery Model B**

After all the checks have been performed, the ProSe Function allocates a ProSe Response Code and a ProSe Query Code associated with a Discovery Query Filter. The UE is then sent both Code-Receiving Security Parameters and Code-Sending Security Parameters and a Discovery Query Filter (instead of an Application Mask), and the ProSe Response Code. The UE is now ready to start monitoring the PC5 interface for ProSe Query Codes matching the Discovery Query Filter.

The Discoverer issues a Discovery Request as well, but specifies it wants to be the Discoverer. The request is checked by the HSS to verify whether the subscriber is authorized to use Restricted Discovery Model B; if so, then the ProSe Application Server is queried to get the PDUIDs for the RPAUIDs and the RPAUIDs which the requesting RPAUID is allowed to discover.

Depending on the HPLMN of the target PDUID, either the current ProSe Function queries the ProSe Function of the other PLMN for a ProSe Query Code and a ProSe Response Code or it already has them. If it does not trust the querying PLMN or the policy for the specific ProSe Application Server demands it. Then the ProSe Function can check with the ProSe Function as to which RPAUIDs the requesting RPAUID is allowed to discover.

A Discovery Response Filter is then created by the ProSe Function based on the ProSe Response Code. The Discovery Response Filter and the ProSe Query Code are sent to the UE that requested to be a Discoverer and it is now ready to send its ProSe Query Code over the PC5 interface.

When the Discoverer has sent a ProSe Query Code that matches a Discoveree's Discovery Response Filter, the Discoveree answers with its Discovery Response Code. Upon receiving a ProSe Response Code, the Discoverer sends a Match Request to the ProSe Function if it does not already have the RPAUID of the Discoveree. The Match Request contains the RPAUID of the Discoveree UE, its own identity, Application ID and ProSe Response Code.

The ProSe Function checks whether the Discoverer UE is authorized to use Restricted Discovery and then finds the target RPAUID from the ProSe Response Code. Optionally, if the ProSe Function does not already have the corresponding PDUIDs for the RPAUIDs or wants to verify that they actually belong to the specific UEs, then it can query the ProSe Application Server for the Application ID. An acknowledgement is then sent to the Discoverer with the Application ID and the target RPAUID.

2.2.4.3 *Direct Discovery for Public Safety use*

Discovery for Public Safety UEs support both model A and model B and the discovery target can either be other Public Safety UEs or UE-to-Network relays. It is a restricted discovery, but the procedures are different from that for non-Public Safety UEs.

A Public Safety UE is provisioned with authorization policies (if it is allowed to use model A and/or B) and radio parameters for use when not served by a E-UTRAN. If a UE cannot locate itself within any geographical region for which it has radio parameters, then it is not allowed to transmit.

In Model A, a UE-to-Network relay announces its presence with a ProSe Relay UE ID that is a link layer ID for the relay and a Relay Service Code to indicate what services it provides. A ProSe Relay UE ID should be unique for every Relay Service Code it uses. Relays only announce and allow UEs to discover it (in Model A), not the other way around (i.e., UEs cannot solicit a UE-to-Network relay). However, UEs can announce to discover nearby group members by announcing its ProSe UE-ID as well as Discovery Group ID. Multiple announcements can be sent if the UE belongs to multiple groups. A UE-to-Network relay can also send a message containing extra information, such as the E-UTRAN Cell Global Identifier (ECGI) of the cell it is served by.

To find other group members in Model B, the discoverer UE sends out a message containing Discoverer Info providing information about the discoverer, a Discovery Group ID, and Target Info providing information about the user or group that is targeted. If the Target Info matches a nearby

UE, this this other UE (a Discoveree) answers with a Group Member Discovery Response message containing its ProSe UE-ID, Discovery Group ID, and Discoveree Info providing information about the Discoveree.

If the UE instead wants to discover nearby UE-to-Network relays the UE announces its Discoverer Info, Relay Service Code indicating what services the UE are interested in and its ProSe Relay UE ID. Relays can then answer with a UE-to-Network Relay Discovery Response message containing ProSe Relay UE ID and Discoveree Info.

In addition to the possibility of a message with extra information in Model A containing the ECGI of the cell the relay is served by, the UE can also request the ECGI from the relay if this information is required by the application.

2.2.4.4 *EPC-level ProSe Discovery*

ProSe also offers a form of discovery resembling present LBS. In EPC-level ProSe Discovery the UEs report their location to the ProSe Function. The ProSe Function determines proximity and communicates with the Application Server to determine which UEs are allowed to discovery each other.

2.2.5 Direct Communication

In ProSe there are two different kinds of Direct Communication depending on what spectrum the UE is allowed to use. Public Safety UEs are allowed to use the spectrum for PC5 to communicate directly even when they are not within any network’s coverage. Other users have to use Wi-Fi Direct instead.

2.2.5.1 *Public Safety use*

Public Safety users can either use PC5 or a UE-to-network relay to communicate one-to-one or one-to-many.

In one-to-many communication, the UE is configured with group information and radio resources are allocated. Since there is no connection, the UE starts listening to the specified radio resource and filters out messages with the correct ProSe Layer-2 Group-ID. All members of a group share a secret that is used to derive a group security key for encryption of all messages.

In one-to-one communication, the UE is configured with a long term key that can be symmetric or a public/private key pair. This form of communication is connection-based and when communication between two UEs is established, lower layer keys are derived to encrypt and integrity protect the communication. A connection is identified by the combination of the Layer-2 IDs of the users, which enables a UE to have multiple one-to-one communication links active concurrently.

If a user is communicating through a UE-to-network relay, it can request the serving ECGI from the relay.

2.2.5.2 *EPC-support for WLAN Direct Communication*

One alternative for WLAN Direct Communication is Wi-Fi Direct, also known as Wi-Fi Peer-to-Peer (Wi-Fi P2P). According to the Wi-Fi Alliance Wi-Fi Direct has a maximum range of up to 200 meters [22]. The EPC can assist in setting up a WLAN direct group based on Wi-Fi Direct. If the ProSe Function decides to trigger the establishment of a WLAN direct group, it sends the necessary Assistance Information to the UEs to enable them to set up this communication. If the UE accepts the information, it sends back a response that can include parameters for the group, e.g. a channel number.

If WLAN Direct Communication was the goal of EPC-level ProSe Discovery, then the necessary Assistance Information was already passed as a part of the Proximity Alert.

2.2.6 Security Aspects

For a discovery message sent over PC5, there are multiple protection mechanisms for security and privacy. Section 2.2.6.1 describes mechanisms applicable to both Open Discovery and Restricted Discovery, while Section 2.2.6.2 and 2.2.6.3 describe mechanisms applicable only to Restricted Discovery. The ProSe Function decides which types of protection should be used when it sends the Code-Sending Security Parameters and Code-Receiving Security Parameters to the UEs in Restricted Discovery.

2.2.6.1 *Message Authentication Code*

In Open Discovery the UE gets a Discovery Key that is used to calculate the Message Authentication Code (MAC) (see Section 2.4.1.2). A specific constant is used for every message type, the ProSe Application Code, and a UTC-based counter. The UTC-based counter has a resolution of 1 second and the least 4 bits of the counter are included in the discovery message in plain text (hence there is a maximum valid time for the message of 16 seconds). The MACs are always checked by the ProSe Function in Open Discovery.

In Restricted Discovery the UE gets a Discovery User Integrity Key (DUIK) unless certain conditions are met (see Section 2.2.6.3). As in Open Discovery the MAC is calculated with the key, a specific constant, a UTC-based counter with the same properties, and then instead of the ProSe Application Code the ProSe Restricted Code is used.

In Restricted Discovery, the MAC may be checked by either the Discoverer UE or the ProSe Function. If the Discoverer UE was supplied the DUIK, then it performs the MAC check.

2.2.6.2 *Scrambling*

Scrambling can be used to avoid tracking of a UE over time. Scrambling prevents this tracking by removing the relationship between discovery messages sent by the same UE. Scrambling can be performed if the UE was supplied a Discovery User Scrambling Key (DUSK).

To calculate the scrambling bit sequence from the DUSK, a MAC is used that is calculated from the UTC-based counter with the 4 last significant bits set to zero and the DUSK as the key. The message and the resulting bit sequence are than XORed together.

2.2.6.3 *Confidentiality*

There is also the possibility to provide message-specific confidentiality for part of the discovery message. This could be used if it is desired to obfuscate part of the discovery message from UEs that are allowed to discover the Discoveree or if multiple UEs use the same DUSK.

Message-specific confidentiality is the last step to protect the message. This is performed by creating an encrypted_bits_mask and a Discovery User Confidentiality Key (DUCK) by calculating a MAC and XORing the output with the message. The encrypted_bits_mask specifies which bits in the message are obfuscated after the operation. This mask is needed to allow for partial matches by the receiver. The MAC is calculated with the DUCK, a UTC-based counter, and parts of the message that are to be obfuscated. This is done in such a way that only the non-encrypted bits in the resulting message are needed when calculating the bit sequence used for the obfuscation, hence the receiver can deobfuscate the message upon arrival.

According to the specification, a MAC is not needed when only one ProSe Code is protected by a DUSK that is matched by the receiver or if message-specific confidentiality is used and the receiving UE does not have the DUCK [21].

2.3 Location privacy

According to the Oxford Dictionaries, privacy is "a state in which one is not observed or disturbed by other people" [2]. Location privacy is therefore the ability to conceal one's location from others or avoid revealing one's location.

There are many reasons to implement techniques to protect the user's privacy. One of these reasons is to fulfill legislative requirements. Another reason is to satisfy users concerned about their privacy. Yet another reason is to protect the creators of services from bad publicity in case of misuse.

A location can be combined with other data and depending on the context a certain combination can be a threat to the users' privacy or not in need of protection. A location can have an ID linked to it that can either be a pseudonym or correspond to a specific person. In addition to an ID, there can also be a time indicating when the user resided at the location.

2.3.1 Legislation concerning Location Privacy

The process of creating new laws is a slow process, hence relatively little legislation is currently in place to protect users' location privacy. In most countries there are laws protecting the user's location information that is obtained by their mobile network operator, such as the Swedish law *Personuppgiftslag (1998:204)* [23] and the EU directive EUR-Lex-31995L0046-EN [24].

In the US there are states with laws to protect user privacy, but so far there is no clear protection of location information on a federal level. There are bills such as the GPS Act, Online Communications and Geolocation Protection Act, and Location Privacy Protection Act that are being considered [25].

However, despite laws to protect users' privacy, many apps include user agreements giving the creators the right to violate it. Jinyan Zang et al. [26] tested 110 popular free apps for Android and iOS and found that 47% of the iOS apps and 33% of the Android apps send location information to third parties. Hazim Almuhimedi et al. conducted a study of the use of a permission manager and an app to give statistics of other apps' permission usage. They found that 98% of the users ended up reassessing their apps permissions and an example of statistics from one of the user was that the participant's apps used the device's location 5398 times with 10 different apps during 14 days [27].

2.3.2 User studies

Many studies have been done about how valuable users think their location information is to them. The users have been offered money or the chance to win something if they give up location data for a certain time. In [28], George Danezis, Stephen Lewis, and Ross Anderson reported that the median price users demanded for a month of their location information was 10£, while in [29] Dan Cvrcek, et al. used more participants and reported a median of 20£ using a similar method. Both studies were done using an auction where the users could offer their location data to a possibly privacy violating study. Participants did not know the study was on how valuable they thought their own location data was. John Krumm offered the participants the chance to win a mp3-player worth

US$200 to give up a month of location data and only 20% of the 97 participants they asked did not want their location data shared with 3rd parties [30].

Another study done by Eija Kaasinen [31] revealed that users did not realize that when using location-aware services they could be located while using these services. This study also showed that the participants were generally not concerned about privacy issues related to location-aware services.

A study by Louise Barkhuus and Anind Dey [32] about both location-aware services and location-tracking services showed that how useful participants found the services affected the perceived intrusiveness on their privacy. It was also shown that location-aware services did not feel as intrusive as location-tracking services.

2.3.3 Risks with reduced location privacy

Janice Y. Tsai et. al. studied the risks concerning location privacy by conducting surveys [33]. The risk that was believed to be most likely was to be bothered with ads based on location, but it was also one of the risks that were expected to cause the least harm. What was believed to potentially cause the most harm was to be stalked. However, this was one of the scenarios that were believed to be least probable. Some of the other risks that were expected to potentially cause a lot of harm were to reveal the location of one's home, being found by someone you do not want to see, or being tracked by the government.

Other risks were mentioned and those considered the most likely or expected to cause the most harm were being spied on by your boss and revealing activities you participate in. There are also a lot of indirect risks related to location privacy such as the possibility for burglars to know based on your location when you are not home or to locate targets (e.g. identifying a defense contractor for social engineering or theft of potentially sensitive data).

2.3.4 Location Privacy Protection Mechanisms

Marius Wernke, et al. [34] briefly describe and evaluate a number of the existing mechanisms based on what protection the mechanisms gives. A big difference between different location-privacy preserving mechanism (LPPMs) is the requirement for a trusted third party. Whether a trusted third party is available or not effectively determines which kinds of LPPMs can be used. Other factors that limit the choice of LPPM are what aspects (ID, location, and/or time) the mechanism is to protect.

2.3.4.1 *Spatial obfuscation*

Spatial obfuscation occurs when the precision of the location is reduced to protect the exact location of the user. In [35], C. A. Ardagna et al. describe the location as a circular area with a certain midpoint and radius in which the user is located with a probability of 1 and the probability density function is uniform over the area. The obfuscation techniques considered are enlarging the radius, shifting the center, and reducing the radius. Their results are that any of these methods can be equivalent with the others. The three methods with appropriate parameters give the same relative privacy.

If coordinates with a certain resolution are used and the location precision is reduced by removing some of the least significant digits, this effectively enlarges the radius and shifts the center in a predictable direction. However, this area is shaped like a square, rather than a circle. The new area is guaranteed to contain the user's real location originally given by the more exact coordinates.

2.3.4.2 *K-anonymity*

K-anonymity is a general method to protect users' privacy. It was first described by Pierangela Samarati and Latanya Sweeney in [36]. To define k-anonymity, they first define quasi-identifiers as a combination of characteristics that can be linked to a user's identity. In addition to quasi-identifiers, explicit identifiers exist (such as names) and these are assumed to have already been anonymized in some way. K-anonymity is based on the principle that given any set of quasi-identifiers, the result should be that at least k indistinguishable users match these identifiers. To achieve this property generalization or suppression could be used for the quasi-identifiers. Examples of generalization are replacing the user's age is by an age interval or replacing an address by a neighborhood. Suppression occurs when information is simply not disclosed.

However, many flaws in k-anonymity has since been presented, e.g. the fact that if all individuals among the k matching the quasi-identifier have a common attribute, then this information is not protected from an attacker that is able to get to the group of k without previous knowledge of the common attribute. The method has since then been extended in many ways to address this issue. Ashwin Machanavajjhala et al. presented the l-diversity method in [37], saying that among the k individuals having the same quasi-identifiers, the sensitive information should have l diverse well-represented values. Ninghui Li et al. further adds to the concept in [38] by stating that any distribution of sensitive data among the k individuals with l-diversity should be close to the distribution of the whole set of individuals. The distance between the distributions should be no larger than t, which leads to the name t-closeness.

To implement k-anonymity for a LBS, either (1) a trusted third party could be used [39], but then all users would need to trust the party or (2) it can be done using peer-to-peer techniques [40]. In [39], Marco Gruteser and Dirk Grunwald implement k-anonymity and calculate the average location accuracy for a user in traffic in a city. With k equal to 5 they get a location accuracy of between 30 and 250 meters.

2.3.4.3 *Position dummies*

To avoid revealing the true location to a LBS, one can interact with it using false location data, referred to as dummies. The provider of the LBS cannot distinguish the real location that results in useful data to the user from those locations that the user simply discards the data from, assuming that the dummies are chosen in a realistic way.

In [41] Hidetoshi Kido, et al. discuss two main issues with implementing position dummies. One of these issues is how to create dummy location data that is indistinguishable from the real location data. Depending upon the type of LBS, the frequency and accuracy of the data is different.

In a LBS requiring continuous location updates, such as a navigation service, the LBS knows how far the user is likely to have traveled within an update interval. If dummies appear at random locations, then they can potentially be identified, thus exposing the real location. It is therefore proposed that a device who wishes to preserve its location privacy keep track of previous dummy locations and generate dummies in the neighborhood of these locations. If the location of other users is known (or can be known), then one should avoid placing dummies in regions with many other users – because evenly distributed locations provide higher location privacy.

A problem with position dummies is that location privacy is increased when the number of dummies are high, but communication cost as well as processing cost in the LBS are also increased. In [41], Hidetoshi Kido et al. considers processing cost to be negligible; however, that may not always be the case. The solution they present is to not treat the location data as a pair of latitude and longitude, but to supply the LBS with multiple latitudes and longitudes for which one combination is the user's real location and the others are dummies. The LBS then has to treat all combinations as real. With this approach the number of combinations functioning as position dummies will be higher than if the pairs were fixed for the same cost of communication. The cost of communication will naturally always be higher than without position dummies if the communication is direct. However, if there is a possibility to have a trusted proxy in between [42], the dummies can instead be generated by it and no communication overhead is introduced in the first step of the transmission, such as over the air interface.

2.3.4.4 *Mix zones*

A mix zone is a concept analogous to mix nodes in anonymous communication. A mix node collects packets and then forwards them in another order. As a result, it is no longer possible to link incoming packets to outgoing packets. Given packets from a sufficient number of users, anonymity will be provided. In the case of location, Alastair R. Beresford and Frank Stajano define the concept of mix zones in [43].

They first define an application zone as a zone where a user has registered an application callback. A mix zone is a zone in which no user has a registered callback. When entering a mix zone there is no longer any need for the user to update the LBS with location information. A trusted system in between the LBS providers and the users can handle the location data and only share this location data with the applications that need it and only do so when the user is in specific locations. The trusted system only reveals the user's pseudonym to LBS providers. The reason a trusted system is needed is to avoid any link between the pseudonym and the user's real identity. Whenever the user is in a zone without any registered callback, then the user can be given a new pseudonym. This is similar to a packet in a network being mixed with other packets, as it is no longer possible to track this user. The trusted system can determine the boundaries for a mix zone before use or do so as the users change their application callbacks. For this method to work there must be other users within the zone, otherwise mixing will not occur. The greater the number of users the greater the mixing that is possible.

In the concept of mix zones, it is assumed that every application can be malicious and that different applications can collude with each other. However, a mix zone does not solve these issues completely, as an attacker can still use other knowledge (such as speed of travel and movement patterns from previous location data) to estimate the probability of the user’s movements within the mix zones and identify users as they reappear outside of mix zones.

2.3.4.5 *Coordinate transformation*

A coordinate transformation is a mapping between the coordinates for a certain point in a coordinate system to the coordinates of the point in another coordinate system. If transformations based on translations and rotations are considered, the result of multiple transformations can still be expressed as a single translation and rotation. If the transformations are bijective as with rotation and translation, then the transformation can then be inverted. The transformation will also be distance and angle preserving when used on a coordinate system based on e.g. a plane. However, coordinates based on the ellipsoid earth can only be considered to have these properties locally.

A coordinate transformation can be used instead of a symmetrical key as the shared secret protecting the location. A difference is that range queries are still possible. Andreas Gutscher describes this mechanism in [44] and points out that there are many possibilities for an attacker to weaken its security. Limiting how the mechanism is used can resist such attempts and having time based transformations could be an option.

2.3.5 Location privacy attacks

An attack on someone's location privacy can have different objectives. Reza Shokri et. al. mentions 3 different types of objectives in [45]. The first objective mentioned is tracking. In the case of a tracking attack, the attacker is interested in reconstructing the user's location trace. A part of this trace can be of interest or the aim may be to have a complete location history of the user. The distinguishing property of this objective is that past sequences of location data are of interest. The second objective mentioned is localization: to find the user's location at some specific time. Here only a specific time is of interest rather than a sequence of locations (as for the tracking). The last objective is meeting disclosure: two users met at some place. The occurrence of a meeting is based on whether the users' location traces have crossed paths somewhere at the same time. The length of time and closeness of proximity could determine the probability of their interaction.

A differentiation Reza Shokri et al. do not make, but could make in the case of localization is to determine the user's current location. Revealing a user's current location could pose a more imminent threat to this user, especially if there is the possibility of physical interaction.

In [34], Marius Wernke et al. describe practical methods to actually attack users' location privacy. The flaw in k-anonymity occurs when all k users share an attribute and the location is the same for all k users. Another flaw in k-anonymity that can be exploited is the need for k users. To satisfy k-anonymity, a user in a sparsely populated area will be grouped with other users further away than users in a more densely populated area. If an area is the result of k-anonymity and it includes both sparsely populated areas and densely populated areas, it is likely that the user that is mixed with the others is one of the users in the sparsely populated areas. If it was not, then the result might not have contained those areas since the nearby users in the densely populated area would have been enough to achieve k-anonymity. Personal knowledge about the user can also be used to single this user out from the crowd, e.g. if the area includes a region that the user is known to visit, then it is likely that the user is located there as opposed to an area where they are not known to visit.

Another approach is to use knowledge about the nearby geography. If the user hides their exact location, then rivers, buildings, and vegetation can be used to determine where it is *unlikely* that the user is located.

2.3.6 Location privacy quantification

Location privacy has been quantified differently in many articles. Depending on the LPPM, the authors quantify location privacy in a way suitable to their mechanism. Using k-anonymity this quantification becomes how many users a given user is indistinguishable from, but using another LPPM this quantification is inapplicable. The same problem occurs with position dummies where quantification can be the measure of how many dummies cannot be distinguished from one's actual location. Despite both quantifications being quite similar, with the only difference being that the other location data is for dummies or other users, a generalized combination of these attempts at quantification does not fit all LPPMs.

In the case of spatial obfuscation, Reza Shokri et al. defined a location privacy metric that is quantified in 3 aspects: accuracy, certainty, and correctness [45]. These are defined as:

Accuracy	$\widehat{Pr}(x\|o)\,, x \in \mathrm{X}$	2-3	The estimate of the probability of x having an observed trace o, with x being a member of the set of attacker objectives. The accuracy is quantified with a confidence interval and a confidence level.

Certainty	$\widehat{H} = \sum_{x} \widehat{Pr}(x\|o) \log \frac{\mathbf{1}}{\widehat{Pr}(x\|o)}$	2-4	The quantification of correctness is defined as the entropy of the distribution of the accuracy. A higher entropy corresponds to a lower certainty.
Correctness	$\sum_{x} \widehat{Pr}(x\|o) \\|x - x_c\\|$	2-5	Correctness is quantified as the expected distance between true result for the attacker's objective, x_c, and the estimate of it. In the cases there is a distance defined between the members of X, then the correctness is given by the expected estimation error in the equation above. Since x_c is what the user wants to hide from the attacker, Reza Shokri et al. considers correctness the metric that determines the users' location privacy. This definition is more general than the definitions based on k and number of dummies. Correctness cannot be evaluated by the attacker since it does not know the user's exact location.

2.4 Security

This chapter describes the necessary security mechanism that are already included in ProSe or could be used in solutions.

2.4.1 Cryptographic primitives

A cryptographic primitive is a fundamental building block used when designing secure systems. These primitives are designed to do a specific task and are assumed to fulfill certain security criteria.

2.4.1.1 *Cryptographic hash functions*

A hash function is a function mapping an arbitrary sized input to a fixed size output. When used in cryptography there are 3 requirements to be fulfilled for a hash function to be considered secure:

- The function has to be one way. If the output is known, it has to be computationally hard to find a possible input for which the hash matches the output. Therefore, given Y it has to be computationally hard to find an X such that $Y = Hash(X)$
- Two different inputs should not give the same output. Therefore, it should be computationally hard given X_1 to find an X_2 such that $Hash(X_1) = Hash(X_2)$.
- It should also be computationally hard to find an X_1 and an $X_2, X_1 \neq X_2$ for which $Hash(X_1) = Hash(X_2)$.

A cryptographic hash function is only based on the target data and therefore only the integrity of the data can be protected if the data's hash is appended to the data. Anyone with knowledge of the algorithm used can modify the data and calculate a new hash value. A message for which only its integrity is protected can be spoofed (i.e., made to appear as if it was from a different sender or for a different recipient). To protect against this the authenticity has to be protected as well.

2.4.1.2 *Message Authentication Code*

To prevent an attacker from being able to create her own messages or modify existing ones, some kind of message authentication is needed, such as using a Message Authentication Code (MAC). A MAC is a one-way function similar to a hash, but based on both input data and a key. The result is a bit sequence that can only have been generated by someone with the key. When the message data is used as input to the function and the MAC is appended to the message, then one can verify that the data has not been changed between being sent and received. The MAC protects both against malicious change as well as transmission errors.

In the ProSe security standard [21], a MAC is referred to as a Message Integrity Code (MIC). While this may have been done to avoid confusion with the use of MAC for Medium Access Control, according to the Vocabulary for 3GPP Specifications [46] this is an incorrect usage. According to this vocabulary, MIC is an abbreviation for Mobile Interface Controller. This means that the interpretation of MAC has to be determined by the context of its usage. These contexts are described as *protocol layering context* and *encryption context*.

The MAC that is used to check the authenticity of a discovery message in ProSe is HMAC-sha256. A HMAC is a MAC that is based on a cryptographic hash function. The extension "sha256" indicates which cryptographic hash algorithm it is based on.

HMAC was presented by Mihir Bellare et. al. in [47]. Before HMACs, MACs were mostly based on block ciphers, but the use of hash functions was motivated by faster software implementations and no export restrictions (at the time) of hash functions. Since hash functions do not use a secret key, its usage has to be defined for use as a MAC. HMAC was designed to be independent of the hash function, thus enabling the HMAC to be used as a black box so that its implementation does not need to be changed, this enabling existing hash implementations (in both software and hardware) to be used.

In IETF Request For Comments (RFC) 2104 [48] Hugo Krawczyk et al. describe the use of HMAC in a more practical sense. The principle of how it is used is specified in equation 2-6. This is the reference used for the implementation in ProSe [21].

$$HMAC - x(key, msg) = Hash - x\big(key \oplus pad_{out}\ CONCAT\ Hash(key \oplus pad_{in}\ CONCAT\ msg)\big) \qquad 2\text{-}6$$

The above requirement ensures that given data and a hash that matches the hash of the data, the hash was with a high probability calculated from the data. The possibility of a larger input than the output makes the 2nd and 3rd requirements for a good hash function leads to limitations that performing certain functions should be computationally hard. To guarantee a function to be collision free is not only hard to achieve practically, but also theoretically impossible according to the pigeonhole principle, as if there are more pigeons than pigeonholes, some holes need to contain more than one pigeon. In addition, a function with an output of n bits has an expected collision after only roughly $2^{n/2}$ tested inputs according to the birthday paradox [49].

In addition to the requirements for a good hash function, a MAC also has to preserve the secrecy of the key independently of the message it is used to protect [50]. For HMAC Mihir Bellare et. al. proved in [47] that the strength of the MAC is essentially the same as that of the hash function used.

Simply using a MAC does not protect against replay attacks, but the use of a MAC is *necessary* to guarantee the integrity of the actual message and therefore use of a MAC is necessary for the replay prevention mechanisms used inside the message. Without a MAC the message could be modified to bypass the replay protection mechanisms.

2.4.2 Replay attacks

A replay attack occurs when a transmitted message is recorded and sent again to trick a user into thinking this is a (new) message they are supposed to process. A replay attack does not require the attacker to be able to read the message or even produce the message. As a result, even an encrypted message can be replayed. Additionally, part of a message might be replaced with an old version of the message, a complete message, or even a series of complete messages that are replayed to achieve the attacker's goal. These messages could even be from another protocol [51].

Usually when referring to a replay attack one means replaying the messages later in time, thus the message is sent again at a later time to produce the same result again at that time. In [52] replay attacks are classified in two main categories: run external attacks and run internal attacks. The former requires the attacker to replay a message from the same run of the protocol. The taxonomy then defines deflections as a message sent with an intended recipient being directed to a non-intended recipient.

2.4.3 Replay attack prevention

Replay attacks are prevented by making every message unique by adding information about the context it is sent in. This context can be described in many ways and if it is not chosen carefully, replay attacks might still be possible.

To add a timestamp in a message can limit the time this message is valid to a specified time period. However, this requires the communicating UEs to have a common time reference and if the clocks are out of synchronization, then the time stamps could appear to be invalid when they are in fact valid. If the time stamps are of sufficiently high resolution, then they can be used as a sequence number and could be used to keep track of which messages in a sequence have been received. To allow messages to arrive out of order and avoid processing the same message twice in case of a replay attack, a sliding window of valid sequence numbers can be used as in the IPSEC standard [53][54] and SRTP [55].

2.4.4 Spatial replay

Spatial replay involves replaying a message intended for *a certain area* (as opposed to a specific recipient) in another area. This is another form of replay attack. This is also a replay in time. Since the message was originally intended to be processed by any number of users and none of them is tricked into processing it twice, this type of attack is quite different from a traditional replay attack. Note that this type of attack is not a deflection, since the original message did not have an intended recipient. Moreover, this replay will be a valid run of the protocol, but with the recipient being in another area than the intended area. Previously used mechanisms for replay attack prevention are not sufficient. As this thesis focuses on spatial replay attacks in the case of ProSe, we will need to carefully consider *what information must be added about the context of the UE(s)* to prevent spatial replay attacks.

2.4.4.1 ***Principle of full information***

In [56], Carlsen suggests that data has to be included to be able to identify the protocol used, the run of the protocol, the current transmission step, message components, and the primitives types of the words. To explicitly type everything would severely limit performance; therefore in [51] Tuomas Aura advocates implicitly typing messages by having slightly different algorithms for message integrity. If the algorithm used for a specific message is unique, then it cannot be interpreted as a message for another stage.

To achieve the above result without developing a new algorithm for every protocol and message type, a known algorithm can be instantiated with a constant instead. This constant can be decided at the design time of the protocol and should be unique to avoid reducing its effect.

The *principle of full information* was introduced by Thomas Y. C. Woo and Simon S. Lam in [57] after a simple authentication protocol designed by them was found to be vulnerable due to a simplification in its creation that did not make it *full information* any more. With *full information* everything that is known by the sender is sent in every cryptographically secured message. By including more information, there are more ways to distinguish a message in a certain context from another message. This principle not only can protect against replay attacks, but also protect against other attacks, such as *a chosen plaintext attack*[*].

Unfortunately having *full information* in every message is not always possible, especially when performance is important, since this information adds a lot of extra data to the message. In [51] it is suggested that only the information known by the sender but not known by the receiver be sent together with a hash of the redundant information known by both. Although this hash does not preserve the actual information, it can be considered a necessary supplement to the data that must be communicated if one wishes to protect against replay attacks. This method of implementing the principle of full information gives a result similar to having slightly different algorithms for every message type as described earlier.

Since a message is already likely to contain a hash or a MAC, this can also be used to implement the *principle of full information* at the cost of only a slight increase in the overhead for the computation of the hash or MAC, while no extra bytes need be added to the transmission. Depending on what is considered full information, some extra memory could be necessary to store the data that is to be included in the computation of the hash or MAC.

Streekanth Malladi et al. comment on the *principle of full information in* [58] and point out that the information in a run can be the same as in another run, therefore a session-id is also necessary. According to Streekanth Malladi et al., this session-id should be based on a random number from every participant; hence no party can trick the others into using the same session-id as has been used before. This could prevent run external replay attacks. The identities of the participants are also mentioned as information; hence these identities could be added when implementing the *principle of full information*.

2.5 Related work

There is little related work to security aspects of ProSe. The work that has been done specifically for ProSe are not concluded yet and no equivalent services exists today. Beacons based on Bluetooth exists that fulfill some of the use cases that ProSe are also supposed to address, but the threat analysis seems to have been completely different (see Section 2.5.1). For the Public Safety specific use, systems such as Terrestrial Trunked Radio(TETRA) exists, see Section 2.5.2, and have been deployed in most European countries. None of the standard for Public Safety communication system address the problem of spatial replay in their standard nor has any work been found discussing it.

In the context of wireless sensor networks and *ad hoc* networks, the problem has been studied extensively and this type of attack is referred to as a wormhole attack. Wireless sensor networks and *ad hoc* networks are different from the communication addressed in the ProSe standard in many ways, hence many of the approaches used with wireless sensor networks and *ad hoc* networks are not applicable. However, related work done in this context is described in Section 2.5.3. However, none of the approaches seems to consider location privacy at the same time as the attacks.

[*] An attack where the adversary can retrieve the ciphertext of any message

2.5.1 Bluetooth Low Energy beacons

Similar technology to ProSe Direct Discovery are beacons based on Bluetooth Low Energy (BLE). An example of a BLE-based beacon is the iBeacon from Apple. It has a range of tens of meters and only supports unidirectional communication. A beacon sends out a 16 byte universally unique ID and 4 bytes to specify region, use case, or similar. Upon receiving a broadcast from an iBeacon, the user can estimate its range and choose to react to it in some way. To deploy devices with iBeacon technology, a license from Apple is required.

There is no guarantee that a received UUID was transmitted by someone with permission from the person or company it is registered to. No security measures are implemented to ensure that an attacker cannot trick a user into thinking they are in the proximity of a legitimate beacon. This in addition to its limited range makes these beacons quite different from ProSe, especially when considering what these different technologies could be used for. BLE-based beacon's main use case has been described as an aid when shopping, e.g. to provide advertisement or guidance when in a specific store [59] or to indoor navigation [60]

2.5.2 TETRA

TETRA is a system designed to be used by Public Safety and other professionals in need of reliable and secure communication. The standard was first published by the European Telecommunications Standards Institute (ETSI) in 1995. It requires a separate network to be used for its communication and its communication speed is low compared to most current commercial mobile networks capabilities.

The security specifications for TETRA do not address spatial replay, but TETRA's more limited use (only the public safety part of ProSe) compared to ProSe does not give as many possibilities. The communication between TETRA terminals and TETRA terminals and base stations is protected by different encryption modes and has some mechanisms to prevent ordinary replay attacks [61][62].

2.5.3 Wireless sensor networks and *Ad hoc* networks

Yih-Chun Hu et al. were among the first to describe worm hole attacks and introduced the term packet leashes to describe a mechanism to prevent it [63]. A packet leash is a limitation on a packet either geographical or temporal for limiting where or when this packet is valid. A geographical packet leash utilizes location information to determine the valid range, while temporal packet leashes uses time stamps to determine the time period when the packet is valid. A geographical packet leash requires the UEs to know their location and to have loosely synchronized clocks, while a temporal leash requires tightly synchronized clocks. Clock synchronization is necessary to bound the possible travel distance of the packet considering that is moving at the speed of light (or possibly at some lower speed when being sent over other media, such as cables or fibers).

Other proposed mechanisms are based on ultrasound, directional antennas, radio fingerprinting, and various methods based on the topology of the network as summarized by Reza Shokri in [64]. Proving proximity can also be done with mechanisms based on utilizing a common radio source [65], using information such as amplitude or phase. These methods will be evaluated in Chapter 4 and, if applicable, customized for ProSe in Chapter 5.

Wormholes has been proposed to be used to protect location privacy. In [66] wormholes are used to hide the source location of messages in combination with random delays in wireless sensor networks. A high privacy level was achieved energy efficiently.

2.6 Summary

This chapter has introduced the reader to Location Based Services and 3GPP's new standard Proximity Services which provides functionality for it to 3rd party application developers. It has also described the concept of location privacy, legislation around it and methods to preserve it when using Location Based Services. It then concluded with security related theory necessary to address the thesis' topic and a presentation of previous work done related to the problem.

3 Methodology

The purpose of this chapter is to provide an overview of the research method used in this thesis. Section 3.1 describes the research process. Section 3.2 describes the attacker and trust model. Finally, Section 3.3 describes the framework selected to evaluate the solutions for preventing or limiting spatial replay.

3.1 Research Process

The research methods considered for this thesis were design science research and constructive research. There is a lot in common between these two methods and the choice could be made based upon preference instead of a balanced choice. Design science does not require instantiation even though it allows it. This might be a better choice since this thesis might get close to a complete implementation, but such an implementation is not one of the goals of this project. Both methods require practical relevance and applicability, but design science offers a more open model. Constructive research is based on pragmatism, while design science is not explicitly based on any philosophic assumptions [67].

Following the design science research method after the requirements have been defined for the artifact that is about to be created, suggested solutions are created. Solutions that show potential will be further developed and evaluated. If necessary, this development and evaluation can be done iteratively as new knowledge is gained. Finally, it will be concluded which solution best fits the problem description. The solutions will as far as possible be evaluated by quantitative metrics, see Section 3.3.

3.2 Attacker and trust model

To perform spatial replay, the attacker needs to be able to listen on the PC5 interface for discovery messages and then, after tunneling them over the internet, broadcast them somewhere else. The attacker is assumed to be able to change the location of its receiving and broadcast units or place them in suitable positions as necessary. An attacker is assumed to be interested in performing spatial replay and is able to locate or track the location of users.

A UE's carrier is assumed to be trusted, but revealing the location to a further extent than already known should still be avoided if possible. The carrier of other UEs is assumed to be less trusted than a UE's own carrier and obtaining the locations of other UEs should only be possible when authorized by each of these UEs.

3.3 Evaluation framework

The solutions will be evaluated mainly by *quantitative* comparisons with evaluation criteria, when possible. The relevant evaluation criteria are: location privacy, changes to the current ProSe specification, processing overhead, and restriction/prevention of spatial replay.

To use ProSe, one has to reveal some information about one's current location. By sending a discovery message this location is revealed to be within 500 meters to anyone that can receive this discovery message. If no one who is not within this direct reception range can learn this UE's location - this will be considered the optimal result with respect to location privacy. For the non-optimal cases and for the optimal as a reference, the quantitative metrics described in Section 2.3.6 will be used. Correctness will be the key measure of location privacy for each solution.

The restriction on spatial replay will be quantified in terms of the maximum distance from the UE which another UE can be *without* the discovery message being recognized as a replay attack. Knowing a given UE's location when less than 500 meters away from this UE will *not* be considered spatial replay, unless there are some other indications of a spatial replay taking place.

To compare changes to specifications is not easily quantified, but could be measured in terms of lines of changes required to the specification. Similarly comparing processing overhead is not easily quantified. One method would be to count the number of bits added to messages over different interfaces and the number of message formats that need to be changed. In addition, any additional computation necessary by different actors should also be considered in this metric.

We will also evaluate whether a given solution imposes any requirements on the UE and the consequences of such requirements, e.g. most solutions will need a location from some source and if that source is GPS we assume that this will require more power consumption by the UE than if a Cell ID or time of arrival based method is used.

When the protection mechanism is affected by the potential speed of UEs, a speed of 30 m/s will be considered as the standard case. It is far from the maximum possible, but it is also probably higher than the average speed of a future ProSe user which could make it a fair balance when comparing solutions.

4 Risk, requirements of a solution and applicable methods

As mentioned in Section 2.5, similar technology does not recognize spatial replay attacks as a problem or even mention it at all. This could be because spatial replay attacks are not considered a threat or in an effort to keep the standard for these others systems simple. However, spatial replay attacks have been considered in wireless sensor networks and in *ad hoc* networks, but the different conditions in the case of ProSe makes most of the solutions developed for these types of networks inapplicable. In this chapter, potential risks are discussed for each type of Direct Discovery. Additionally, the chapter will present what the ProSe standard today has done to prevent spatial replay attacks. Section 4.1 examines the risks of spatial replay attacks in Open Discovery, Section 4.2 examines the case of Restricted Discovery, and Section 4.3 examines the case of Public Safety use. Section 4.4 considers the applicability of security-related methods from related work. Finally, Section 4.5 examines location privacy-related methods from Chapter 2.

4.1 Open Discovery

Replay attacks are generally prevented by the UTC-based time stamp added in every message. Only a few of the least significant bits are added in plain text, but the rest are also used to calculate the MAC and that gives a message a certain validity time. The resolution of the UTC-based timer is a second.

In Open Discovery, 4 of the least significant bits of the UTC-based timer are added in plain text which gives a validity time of 16 seconds. During 16 seconds a UE can move up to around 500 meters if we assume a speed of around 30 m/s (as motivated in section 3.3). Given the 500 meters of ProSe range, this results in the Monitoring UE and Announcing UE potentially being up to 1000 meter apart from each other when the message's validity timer is up. The same limitation also applies in the case of a single attacker recording discovery messages and then traveling at this same maximum speed to another location and replaying them. A reasonable protection against a replay attack based upon tunneling would be to match the limitation of such a physical replay attack. Unfortunately, the captured version of the message could move at roughly the speed of light, hence in 16 seconds the message could have traveled 4.8 x 10^9 meters (an interplanetary distance!).

In Open Discovery the Announcing UE allows anyone to discover it. Users of Open Discovery could be a restaurant or other business for whom location privacy is not important, but rather the opposite, a restaurant or other business might prefer that potential customers know its location. However, for a user with tickets to an event she cannot attend and therefore wants to sell them, knowing that they are within 500 meters of the event might be sufficient to establish contact and negotiate a price, and then the two parties could exchange more precise location information to actually exchange the tickets for a payment.

A taxi company utilizing ProSe that is the target of a spatial replay attack by a competitor could find their reputation ruined if their cars that appear to be nearby repeatedly turn out to be far away. A customer that discovers a taxi and requests a rapid pickup might be told there has been an error and that the closest car is over 10 minutes away, hence they might use another taxi service not only this time but the next time they need a taxi. As we can see, Open Discovery may require both location privacy and spatial replay protection, as sending everything in plain text that is available to all other users could lead to an attack.

4.2 Restricted Discovery

In Restricted Discovery 8 of the least significant bits of the UTC-based counter are added in plain text, thus giving a validity time of 256 seconds. If the same assumption is made for Open Discovery (i.e., that the UEs are traveling at most 30 m/s), then with the addition of the 500-meter ProSe range, these UEs could be approximately 8000 meters apart before the message's validity time is up. This is a much looser requirement than for Open Discovery, thus leading to a greatly increased risk of a single attacker performing a replay attack.

Since Restricted Discovery can potentially be obfuscated the ProSe Restricted Codes, targeted attacks will be harder to perform. Restricted Discovery also has Model B which requires a response message. Therefore, to successfully perform a spatial replay attack requires both the query message and the response message to be replayed. A spatial replay attack would be limited to fewer users; hence the impact of an attacker would be harder to notice.

In the case of discovery between friends or colleagues, there is less risk for a ruined reputation, but rather the users are more likely to decide that the service they are using is unreliable. A service will be needed to give them permission to discover each other and they will both need to be registered in order to do so. Location privacy could be important since individuals will be using such a service, not only companies. This suggests that both location privacy and spatial replay protection are needed. The lower risk of a successful attack may slightly reduce the need for protection from spatial replay. Since Restricted Discovery involves response messages, an announcing UE could vary its transmitting power and determine the range to other UEs by when they are able to respond. Moreover, this discovery would be limited to only those users who are permitted to discover each other.

4.3 Public Safety use

Direct discovery for public safety uses the same validity timer as Restricted Discovery, hence the result is the same. A big difference between public safety use and other uses is that location privacy may not be considered to be as important as in other settings. Pare of the reason for this reduced concern about location privacy is that public safety workers would only reveal their location to other public safety workers who are allowed to discover it. Moreover, to reveal their location in this case should not pose a threat unless another UE is compromised, but not yet blacklisted.

4.4 Applicable security methods

All solutions will be limited by the capabilities of the UEs to be used. A reasonable assumption is that nothing an average smartphone does not have today can be required by any practical solution. Therefore, some of the methods used in related work can be immediately dismissed.

For example, a mobile phone does not usually have the capability to send and receive ultrasound. However, there are applications today of devices that ascertain their proximity by using audio signal of less than ~20 kHz (see for example, both the Asus and TP-LINK OnHub access points can be configured with an Android phone via audio signaling). Another type of solution that does not need further consideration are those that require directional antennas. At present this is something the typical smart phone does not support, although this might be something that will soon appear because of the increasing use of Multiple Input-Multiple Output (MIMO) radio interfaces. Such radio interfaces are widely used with WLAN access points. Therefore, solutions similar to those in related work based on either ultrasound or directional antenna are not considered further, but clearly should be part of future work.

A solution that is based on radio fingerprinting of devices can clearly be considered. Radio device fingerprinting is capable of identifying a transmitter based on its unique characteristics that affects the emitted waveform. Today a typical smart phone cannot perform radio fingerprinting of other UEs, but the network could potentially detect devices broadcasting Application Codes/Restricted Codes with an incorrect radio fingerprint.

For a common radio transceiver to be used to prove proximity, obviously a common radio source has to exist. For a service targeted to be usable in any country the possibilities for frequency, maximum emitted power, and modulation & coding are rather limited. For phase information to be used the frequency would need to be very low to prove proximity with the intended ProSe range. In [65] successful proximity detection results are achieved when the UEs are fractions of a wavelength (one of the distances considered is 1/200 of the wavelength) apart. No assumptions can be made about common radio sources except for the cellular network when within such a network's coverage.

Another method that can be used is packet leashes. As described in Section 2.5.3, 2 types of packet leashes exist: temporal and the geographical. For a temporal method to work the UEs need a tightly synchronized time reference and need high precision. In ProSe the clocks have to be within MAX_OFFSET from the time supplied by the network, but MAX_OFFSET is not defined in the standard. However, since the UTC-based counter used for ProSe has a resolution of a second this method is infeasible. An alternative could be to force the clock source to be from a high precision source (such as GPS) and the timestamps could be in microseconds rather than seconds. A solution based on temporal leashes is described in Section 5.3.1.

A geographical packet leash could be implemented without changing the clock requirements. However, methods that rely on location information need a reliable source of such information. Coordinates obtained by GPS will require a substantial amount of battery power if the UE is moving and needs constant location updates and possibly some method is needed to determine if it is not moving and hence is not in need of updates. A stationary UE could be preprogrammed with its coordinates so that it does not require a GPS receiver to learn its current location. Moreover, a fixed device could be powered by the electrical grid instead of a battery. Other types of location information include Cell ID for cellular devices and Media and Access Control address for Wi-Fi equipped devices. Geographical packet leashes are presented in both Sections 5.1 and 5.3.

All of the different types of ProSe Direct Discovery are limited by the permissions stored in the HSS. These permissions can be for specific to particular PLMNs, hence if a spatial replay attack is attempted cross PLMNs, then this attack will fail if a UE does not have the required permission to announce the Application/Restricted Code in the PLMN where the message is replayed. If the Monitoring UE or the Discoverer UE is supposed to determine the UE's proximity, then location privacy and prevention of spatial replay will have to be balanced since both of them cannot be achieved at the same time. Since the direction of the Monitoring UE is unknown, attempts at location obfuscation will be very limited.

A solution to the above issues must both provide sufficient protection against replay, while at the same time protect the user's location privacy to a reasonable extent. The solution should avoid add too much overhead to the protocol or limiting non-malicious users of the services.

A flexible solution that allows multiple options could adapt if the threat level is perceived to be different in the future. For example, if the threat proves to be a severe vulnerability, then the prevention of spatial replay could be stricter, while the opposite would reduce the amount of processing needed. An initial solution needs to be possible to implementable today, hence it cannot require a lot of testing to determine suitable requirements for such an initial solution.

4.5 Applicable location privacy methods

Upon receiving a discovery message, the receiver can, without any replay present and in the absence of boosted transmitting power, conclude that the sender is within a 500-meter radius. This is an unavoidable intrusion into the users' location privacy, as this operation is necessary for the services to work. This basic functionality represents the best case after adding a protection mechanism against a tunneling attack. For a temporal packet leash this requirement will be fulfilled; while for a geographical leash, location privacy needs to be considered.

Position dummies are only suitable when there is an untrusted service that the location is shared with and the extra location data does not introduce any negative effects on the user's experience. In the case of ProSe, the other users receiving a position dummy would not know what to do with it. Ideally they will not know it is a position dummy, thus they will conclude that the sender is in proximity; but if it is not, then this approach introduces the same problems as does spatial replay. There would also be the problem of sending to UEs who are not in proximity and therefore out of range. It could be implemented in combination with a geographical packet leash, but this approach would only increase the data available to the other UEs to figure out the sender's actual location. Averaging and other statistical methods could be applied to the location data to estimate the sender's location, especially since the algorithm (and the resulting expected distribution of dummies) could be known to an attacker.

In the case of k-anonymity, its implementation is not obvious. However, its general requirement of a trusted third party does not fit with the communication model of ProSe Direct Discovery. All discovery messages are exchanged directly between UEs and achieving k-anonymity would not be possible without changing a lot of the standard and even so the end result would likely not be desirable. If a number of local users with broadcast indistinguishable codes, this would provide k-anonymity, but would also ruin the service since the user would not know what (whom) was discovered.

In conclusion, this leaves coordinate transformations and spatial obfuscation. Coordinate transformations could protect users' location privacy from other carriers in EPC-level Discovery, but general purpose encryption would probably be a better choice to protect potential location information in a discovery message when the MAC is checked by the ProSe Function. Spatial obfuscation is probably needed for most cases if location information is added in the discovery message, both when (1) the MAC is checked by the receiving UE and no encryption is used and (2) when MAC checking is done by the ProSe Function to protect location privacy from other carriers.

5 Possible solutions to prevent spatial replay in ProSe

This chapter presents potential solutions to prevent spatial replay in ProSe. These solutions are divided into categories depending on where the security is added. Section 5.1 describes solutions in which the location is reported in some way to the network and checked during a Match Report. These solutions require (1) an Application code to be unique and (2) linked to a specific user. Section 5.2 describes solutions in which the network monitors radio resources to detect spatial replay. Finally, in Section 5.3 solutions based on a change in the discovery message are presented.

5.1 Permissions/network based

Changes to the ProSe message flow could be made to limit spatial replay. Depending upon the nodes chosen to be involved, the amount of extra network load needed will be different. All of the solutions in this section utilize the network, as they only work when there is network coverage, and all of these solutions require (1) an Application Code to be unique and (2) linked to a specific user. However, these two requirements are *not* specified today, hence they would need to be changed or explicitly stated.

For most of the solutions in this section, the ProSe Function requires the monitoring UE's location. One of the many means to learn this location would be for the monitoring UE to add its location in the Match Report. Another option would be for the eNodeB that the UE is communicating via, to estimate the UE’s position based on its own known location. A third method would be to use U-TDOA, but this would require the use of several nearby eNodeBs.

When the ProSe Function determines proximity based upon a Match Report, it has to take into consideration the UE’s speed. If a UE is moving rapidly it could now be at a different location than when it last updated the ProSe Function with its location. If this distance is too great, then spatial replay will still be possible; therefore, the update frequency needs to be sufficiently high that the distance that this UE could have moved between updates is less than 500 m.

5.1.1 Permissions

When a UE wants to announce, it first contacts the ProSe function to request permission for making an announcement. This permission is currently given for a specific PLMN, but this permission could be changed to be valid only for a certain coverage area served by this specific PLMN. If so, then an announcing UE would need to update its permission every time its location changes (i.e., after it enters another area). According to the current standard, these permissions are stored in the HSS, hence making such a change would involve the ProSe function since it is in contact with both the UE and HSS.

5.1.2 Tracking areas

To reduce UE paging overhead, location areas exists in GSM/UMTS and similar tracking areas exist in LTE. These location areas could be used for ProSe. The areas are set up by the operator who decides how large each area should be and which BTS/NodeB/eNodeB should belong to a given area. In GSM/UMTS, each time a UE moves from an area to another, it notifies the network. However, in LTE a UE only notifies the network if the new tracking area is not in the list of tracking areas it is allowed to move between *without* updating its location. A tracking area update is made to the MME by the UE when necessary.

Since ProSe is supposed to be used in LTE, the increased uncertainty given by the reduced frequency of area updates compared to GSM/UMTS will negatively affect this solution. To use tracking areas instead of specific areas for ProSe would require a lot less extra load on the network, since the message flow for tracking area updates already exists (to a large extent) and is in frequent use. An addition would be that a request from the ProSe function to the MME would be required each time a MAC is about to be checked to confirm that the location of the announcing UE is within the same or a bordering tracking area. This would facilitate the use of the tracking areas and the UE's location updates being coupled with the permissions described in the Section 5.1.1.

An advantage of this approach is that it would not require any extra information about the monitoring UEs' locations since this location is already known to be within its tracking area, but a more exact location of the monitoring UE could give a better result if the tracking area is large and the other UE is not in the same or an overlapping tracking area. However, if the distance is too great then there is the potential for a spatial replay attack.

A problem might be that tracking area layouts would need to be shared between operators and this may not be something they want to share. Operators would also need to update each other about changes in these areas, either explicitly if the actual tracking area identifier is used or when sharing data between operators the tracking area identifier could be replace by an area specified in terms of WGS84 coordinates. Note that when the permission is granted the UE's tracking area could be changed to match the desired area where this permission would be valid, hence only those UE's would need to have more frequent area (and location) updates. This also means that the tracking area could be changed to an area meaningful for the ProSe Functions that are to be involved and thus the tracking area need not correspond to the operator's tracking areas for the purposes of paging.

Another problem is that there is currently no direct link between the ProSe Function and the MME. As can be seen in Figure 2-1 on page 9, the only path from the ProSe Function to the MME is through the HSS. This suggests that a new reference point is probably needed to directly connect the ProSe Function and the MME in order for this solution to work well.

5.1.3 SLP

In EPC-level discovery, the SLP node handles the UE's location. It requests UEs to send their location with a certain frequency. This could be extended to also apply to announcing UEs in Direct discovery. The SLP has several options for specifying location accuracy and frequency of updates. Locations are specified using GAD-shapes when interacting with the SLP, see Section 2.1.3, but locations could be coded more efficiently if necessary. GAD-shapes allow for an *unnecessarily* high precision in location. When a MAC is about to be checked, the ProSe function should query the SLP for the announcing UE's location.

5.1.4 Dynamic metadata

Another solution that would partially reuse existing message flows is to utilize dynamic metadata. Dynamic metadata was described in Section 2.2.4. For each application ID, there is the possibility to have metadata and this metadata can be dynamically updated by the UE. The announcing UE could add its location to the metadata and either (1) when the MAC is about to be checked, the ProSe function would read the metadata or (2) if the MAC check is passed, then the metadata can be used by the UE, since this metadata is delivered to the UE.

In addition to performing the checks and adding the UE's location to the metadata, the only change necessary would be to the increase the frequency of sending metadata. The metadata version is specified in the Application Code and a mask is used to obtain the bits used for this data. If there are too many updates, then the version number would need more bits than available. Thus a wraparound will occur; hence, this wraparound will need to be handled. For a stationary UE, such as one located (e.g. in a restaurant) the location will probably not change, hence the overhead will be reduced to the extra data added due to inclusion of the location in the metadata.

5.1.5 Match Report

When a Discovery Report is received by the ProSe Function and the Application ID was issued by another PLMN, then the ProSe Function itself sends a Match Report to the ProSe Function in that PLMN. This has to be done every time in Open Discovery, while in Restricted Discovery this is optional. The ProSe Function that issued the Application ID could detect some attempts at spatial replay by estimating the distance between UEs that sent Match Reports. If the Cell ID of the bases station that the UE is connected to is not already available to the ProSe Function, then the Cell ID or coordinates could be added to the Match Report sent by the UE over PC3. If spatial replay is suspected, then the MME could be queried for the location of every UE allowed to broadcast an Application Code corresponding to the observed Application ID. With only small changes to the current version of the standard this could prevent some spatial replay attempts.

5.2 Monitoring discovery messages

The solutions described in this section are based on having some node in the network capable of monitor the spectrum for PC5 traffic.

5.2.1 Detection of multiple usage of announced code

All of the eNodeBs that offer spectrum for PC5 could monitor it to discover Application Codes that should *not be present* or are *unlikely*. A UE only receives permission for a specific PLMN and as long as it stays within this PLNM, no renewal is needed before the validity timer has expired. When an Application Code appears that has not previously been broadcast in the area and this UE has not acquired permission from this specific base station, then the base station could query the ProSe Function to find out if the Application Code was also broadcast somewhere else.

In a city the distance between eNodeBs can be assumed to be small, hence a shift in the serving eNodeB could happen very frequently which would generate a lot of traffic to check the appearance of new Application Codes. However, this traffic will only be on the links interconnecting the eNodeBs and within the core network – these links already had to carry extra traffic to support the handovers between the different eNodeBs. In contrast, for a sparsely populated area the distance will be larger between the eNodeBs and it might not be able to monitor the whole area it covers due to the low emitted signal strength allowed for ProSe.

No message flow exists today in the standard to check if an Application Code is used somewhere else or not. Since an Application Code can be broadcast in different PLMNs and in different parts of the world, to implement a solution similar to this would require most eNodeBs to communicate with each other and would scale extremely badly in the real world. However, the scaling could be improved by first processing this data within a given PLMN and then communicating the relevant information to the other PLMNs in the area where this specific Application Code is valid.

5.2.2 Device radio fingerprinting

Each radio transmitter has unique characteristics and these could be used as its fingerprint. A mobile phone does not typically have the ability to fingerprint a transmitting device, but a node in the network could have this ability. When a UE requests permission to announce a ProSe Application/Restricted Code, the UE's radio fingerprint can be obtained by the network and be stored as a valid radio fingerprint for this specific code. The spectrum for PC5 can be monitored and the radio fingerprints of UEs' broadcasts can be checked. If a code is broadcast with a device fingerprint that is not a UE that has been permitted to broadcast this code, then the owner of the Application ID or the application server could be notified and future Discovery Reports for the code could generate a warning about a potential spatial replay or the discovery could be invalidated.

This solution would cover most users since the distance between eNodeBs are small in areas with a lot of users. However, not all attempts at spatial replay would be prevented even though the attacking UE is in range of a eNodeB that is actively monitoring PC5 transmission - since radio fingerprints are not as unique as human fingerprints [68]. In Open Discovery, the attacker's transmitter could be chosen to match that of the Announcing UE; but in Restricted Discovery due to obfuscation the specific messages cannot be filtered out, hence everything would have to be replayed by an attacker.

5.3 Changes to discovery messages

For most of the solutions in this section, monitoring each of the UEs' locations is required and if the MAC is checked by the ProSe Function, then the ProSe Function also needs to know each of the UEs' locations. As presented in Section 2.1.2, there are many means to get this location.

If the location is retrieved by the UE, then this location can be transferred to the ProSe Function in the Match Request. Either the UE's location can be included with all its bits or just enough bits sent in order for the ProSe Function to uniquely identify this UE knowing its own location or the location of the eNodeB that this UE is connected to. A method that would not require any changes to the Match Request is to use U-TDOA, but this would result in an increased amount of communication between eNodeBs for every Match Report.

A solution based on the UE retrieving its own location without involving the network and adding this location information to the Discovery Message would work both when there is network coverage and when there is not.

The changes to the Discovery Message can be classified in to 3 categories: explicit, implicit, and mixed. An explicit solution would add the information by adding data bits to the message, while an implicit method would only add extra information to the input to the MAC calculation. An explicit method should also need to add this information to the MAC calculation in order to prevent these bits from being altered (without detection) by an attacker. A solution based on temporal packet leashes is described in Section 5.3.1. Following these explicit solutions are described in Section 5.3.2. Finally, implicit and mixed solutions are described in Section 5.3.3.

5.3.1 Temporal packet leash

A temporal packet leash already exists in ProSe, but its purpose is limited to preventing replay attacks in time (where this period of time is quite long). The 16 second validity of a message effectively limits the possibility of an attacker UE replaying the message later when it is no longer meant to be received. Unfortunately, a packet can travel very far in 16 seconds and with a UTC-based counter with a precision of one second, this distance cannot be reduced without first changing the temporal precision of this counter.

In a microsecond a packet can travel up to 300 meters; therefore, a temporal packet leash would require a UTC-based counter with a precision of a microsecond or a fraction thereof in order to be able to limit spatial replay. Optimally, the timestamp would only allow for a few microseconds to pass before the packet becomes invalid, but this would require changing a lot of the assumptions of ProSe. Additionally, it would put very tight time bounds on the processing of ProSe messages or high precision timestamping of these messages by the radio interface.

The Access Stratum, handling the lower network layers, is defined to handle packets transparently of their content. More specifically, it chooses which time slot it sends a packet in. To be able to calculate the MAC and timestamp the message correctly, either the higher layers need to know exactly when the packet will be sent or a second packet can be transmitted specifying what time the previous one was actually sent at (this is similar to how Precision Time Protocol works [69]).

The time source would need to be synchronized between all UEs. Microsecond synchronization is not easily achieved and would be limited to having an on-chip clock with enough accuracy (this would be expensive today) that could be regularly synched or continuously retrieving the time from GPS or another source[*].

5.3.2 Explicit

The explicit method can utilize different location identifiers. In Section 5.3.2.1 coordinates are considered as the location identifier and in Section 5.3.2.2 the Cell ID is used. In Section 5.3.2.3 different solutions to protect the user's location privacy are considered, such as encryption of the location identifier.

5.3.2.1 *Location identifier based on coordinates*

Coordinates could be added to the discovery message. These coordinates could be encoded in GAD-shapes. To avoid revealing the sender’s exact location, some kind of spatial obfuscation could be utilized. Possible methods for this spatial obfuscation are randomized selection of a location close by within a given radius or rectangle. Another alternative is to truncate some bits of the location information. The results of this spatial obfuscation on location privacy were evaluated in Section 6.2. When coordinates are present in the discovery message, the receiving UE can directly determine if it wants to have the MAC checked or not, thus attempts at spatial replay are stopped at the earliest possible stage.

The Application Code can be composed either of a single bit sequence or a prefix and suffix. When the Application Code is split into a prefix and suffix, this is referred to as an Application Controlled Extension and the suffix can be chosen by the application. This extension is available in both Open Discovery and Restricted Discovery, but not in Discovery for Public Safety Use. Qualcomm has suggested the suffix be used to specify one’s location, thus limiting spatial replay. They suggest 16 bits could be used for the latitude and 16 bits for the longitude. This would give a resolution of 300 meters for latitude and 600 meters for longitude at the equator (or 300 meters at latitude of 60 degrees). See Equation 5-1 for more details.

5.3.2.2 *Cell ID as location identifier*

The discovery message could be extended with a Cell ID. The announcing UE could add the Cell ID of the cell it is connected to or the ID of the cell providing spectrum for ProSe. The size of a cell can range from less than a kilometer to tens of kilometers depending on the density of expected users in this cell. In this way a monitoring UE can directly determine if the origin is sufficiently local for it to accept the message before sending a Match Report. A Cell ID will always be available when the UE

[*] Note that GSM, UMTS, and LTE base stations already have GPS receivers or other sources for high precision time – as they use this information to perform both time synchronization and to control their frequency synthesis (so that they transmit with the correct frequencies). See the Master’s theses by Elham Khorami [70] and Mozhdeh Kamel [71].

has network coverage and it does not require much extra processing to acquire (as the UE is already listening for changes in Cell ID to decide whether it needs to do tracking area update or not). However, this method cannot be used if there are no base stations close to the UE. Fortunately, in nearly all of the locations where there are lots of users there are lots of base stations – hence in practice this limitation should not be too severe. Moreover, in locations where there are no base stations there is limited opportunity for spatial replay.

The Cell ID would have to be chosen from an authenticated eNodeB to avoid an attacker announcing cell information in order to trick users into thinking they are close to an actual cell with a certain Cell ID. The Announcing and Monitoring UE are not necessarily connected to the same eNodeB, thus the Monitoring UE would need to authenticate the eNodeB used for the Cell ID by the Announcing UE. If not, then the serving cell can provide information about valid cells in its proximity. It is unclear whether the cells handling the available spectrum will be named by ID in the information message sent by the serving cell. If they are not, then the cell's location for each Cell ID could be looked up in a database.

5.3.2.3 *Encrypted or transformed location identifier*

To include a location identifier in plain text would reduce the location privacy of the user. To preserve this location privacy either encryption or a coordinate transformation[*] could be used. The key the announcing UE is supplied to check the MAC could also be used to encrypt the location identifier or transform the coordinates. Note that using encryption would give the receiving UE the possibility to coarsely filter out spatial replay attacks, *before* doing a Match Report.

Alternatively, the key could be used to apply a coordinate transformation as described in Section 2.3.4.5. The coordinate transformation should in addition to the bit sequence also be affected by the UTC-based counter. The coordinates that the coordinate transformation has been applied with could be appended to the discovery message and announced by the announcing UE. When the monitoring UE receives a discovery message, it could also have the same bit sequence supplied by the ProSe function along with the discovery filter. Potential spatial replay could then be filtered out directly before wasting any additional resources. Another option would be for only the ProSe function to know the transform, then it could check this location at the same time as it checks the MAC. The ProSe function would then need the monitoring UE's location to calculate the exact distance or its own location could be used to determine if it originated locally.

Alternatively, the location identifier could be obfuscated similarly to how the discovery message is obfuscated in restricted discovery, see Section 2.2.6. Again either the monitoring UE could have been supplied the key or the obfuscation could be reversed by the ProSe function when it does the MAC checking. Either coordinates or Cell IDs could be used as the location identifier.

5.3.3 Implicit and mixed

When calculating the MAC, some kind of location identifier can be added that is not present in the plain text data. This is referred to as implicit and a mixed approach is then possible when some but not all of the bits are included in plain text. The mixed approach can be considered to be an implementation of the full information principle with a hash/MAC as mentioned in Section 2.4.4.1. For example, the location identifier could be coordinates mapped to a grid, while the cell identifier could be added to the MAC. The entity checking the MAC must then know the receiver's location with sufficient accuracy to be able to estimate the sender's position and use this information when checking the MAC. In the case of a grid, if carefully chosen, the processing overhead due to not

[*] In this thesis, coordinate transformations were mainly considered to preserve location privacy in EPC-level Discovery, but since it was eventually left out of the thesis it could instead be considered future work. It does not provide much benefits over traditional encryption

knowing the sender's exact location can be reduced by minimizing the number of calculations of the MAC.

A coordinate grid spanning the earth with equal length and width in longitude and latitude will not be equal in distance on the ground. Since the earth's surface is an ellipsoid, a longitude degree will represent fewer meters closer to the poles. A grid will have to take this into account to provide protection against spatial replay independent of the location where it is used. The circumference of the earth for a given latitude is given by the formula in Equation 5-1.

$$C_{Latitude} = \cos(latitude) * C_{equator} \qquad 5\text{-}1$$

In the discussion that follows the types of grids considered are multiple cylindrical projections with different radii filled with aligned rectangles, non-aligned rectangles, and hexagons. Different sizes of the cells and radii for the cylindrical projections are considered. A few grids created for other purposes are also considered.

The Euler characteristic (χ) describes a relation between a closed surface's number of edges (E), vertices (V), and faces (F). This value is calculated according to Equation 5-2.

$$\chi = V - E + F \qquad 5\text{-}2$$

A sphere has a Euler characteristic of 2. It can then be calculated that the closest to a sphere one can get with rectangles are a cube, while hexagons cannot form a closed surface similar to a sphere at all. If complemented with 12 pentagons it can form a sphere-like shape though, but to map coordinates to it is a quite challenging task. No description of such a mapping has been found and it is assumed to require too much calculations to be of interest. Therefore, poly-cylindrical projections are used in Sections 5.3.3.1, 5.3.3.2, and 5.3.3.3.

5.3.3.1 *Aligned rectangles*

The straight lines on the top and bottom of the rectangles make it easy to change the mapping between coordinates and cells when the latitude changes. Figure 5-1 shows what a grid based on aligned rectangles might look like. In this example the sides are the same length as twice the ProSe range and the cells are labeled with parts of a potential latitude and longitude coding. The receiver is marked with a small dot and a circle around it represents the ProSe range.

A receiver, aware of its own location, can conclude that if it is in the lower right part of the cell, then the sender must be in either the same cell or one of the three closest cells. This yields a worst case of calculating the MAC four times before finding a match. Alternatively, the sender can include sufficient information in plain text for the receiver to be able to identify the correct cell. To distinguish between the alternatives, two bits are necessary and a choice for these two bits could be one bit of latitude and one bit of longitude. As can be seen in Figure 5-1 the least significant bit of each will uniquely identify the cell if they are included.

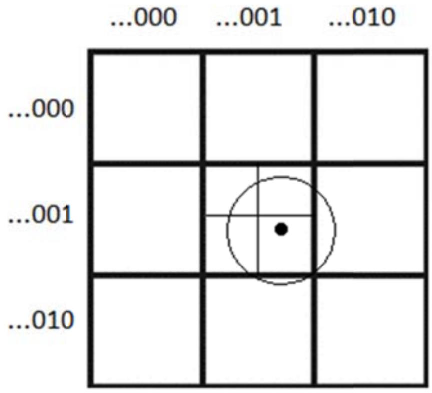

Figure 5-1: A group of aligned rectangles

The mapping between coordinates and the grid's cells can be done by simply truncating a certain number of least significant bits from the longitude, depending on the latitude. The latitude can have a fixed number of bits since it will not change due to the curvature of the earth and 13 bits gives a resolution of 1.2 km. The longitude and latitude can be encoded in the same way as in 3GPP GAD-shapes, see Section 2.1.3. Thus Longitude: $N \leq \frac{2^{bits}}{360}\mathrm{X} < N + 1$ and Latitude: $N \leq \frac{2^{bits}}{90}X < N + 1$ where N is the binary encoded number and X is the latitude or longitude in decimal degrees. The number of bits in Table 5-1 indicate the number of bits, including the sign bit, that would be used to encode the longitude. Latitudes over 84 degrees are unlikely to be used, but one more step has been included in the table anyway. All latitudes over a certain limit could be encoded as one big rectangle, or rather one big spherical cap.

Table 5-1: **Non-zero longitude bits in grid coding**

Latitude degrees	Bits	Longitude resolution
[0, 35]	15	1-1.2 km
(35, 65]	14	1-2 km
(65, 78]	13	1-2 km
(78, 84]	12	1-2 km
(84, 87]	11	1-2 km

If a more adaptable solution is determined to be necessary, then instead of truncating Least Significant Bits (LSBs), the higher values could be left unused. This could be achieved by adding more cylinders to the poly-cylindrical projection (see Figure 5-2) and for every segment the latitude with highest absolute value could be inserted in Equation 5-3 to calculate the number of grid cells in the X-direction. This approach for a grid will be referred to as aligned rectangles 2 in the following sections.

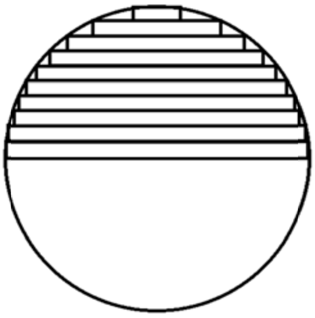

Figure 5-2: **Poly-cylindrical grid**

In equation 5-3, $\emptyset$ is used as the circumference of the earth and θ as the latitude with the highest absolute value. The Width (W) parameter can then be used as the desired grid cell size in the x-direction. To avoid having grid cells crossing the boundary between latitude segments, the grid cell size in the y-direction could be chosen to be an integer number to fit between the equator and the pole. If implemented correctly this avoids the choice of floating point representation from affecting the result. The receiver has to decode the y-direction first to determine the number of grid cells in the x-direction.

$$X_{max} = \frac{\lfloor \emptyset * Cos(\theta) \rfloor}{W} \quad 5\text{-}3$$

The grid cell identifier can then be represented explicitly in the message by any number of LSBs and the full identifier can be included when calculating the MAC. The resulting grid cells should be numbered from 0 at the south pole and increasing in the y-direction and from 0 in the x-direction from 180 degrees West to avoid problems at the equator when using only LSBs. As noted previously the receiver has to decode the y-direction first to determine the amount of grid cells in the x-direction.

To avoid calculating the cosine of values in a real implementation the number of latitude segments could be chosen as a small enough number to be stored as constants instead. This also results in a freer choice of grid cells in the y-direction than if fewer latitude segments are used. However, it reintroduces the size difference that was solved by using a pseudo-cylindrical projection when the latitude segments size approaches 0. The resulting compromise can be seen in Table 5-2.

Table 5-2: Size distortion in a poly-cylindrical grid

Degrees	Size distortion	Number of constants
1	3.1 %	90
2	6.5 %	45
3	10.1 %	30
5	18.3 %	18

Depending upon the size of the grid and the expected inaccuracy of the location of the UEs the number of necessary LSBs can be calculated. In Equation 5-4 a formula is given with B as the number of LSBs, R is the approximate ProSe range, and I_a and I_b are the inaccuracies of UE_A and UE_B.

$$\frac{2^B - 1}{2} > R + I_a + I_b \tag{5-4}$$

Equation 5-4 is derived from Figure 5-3 and generalized to an arbitrary number of LSBs.

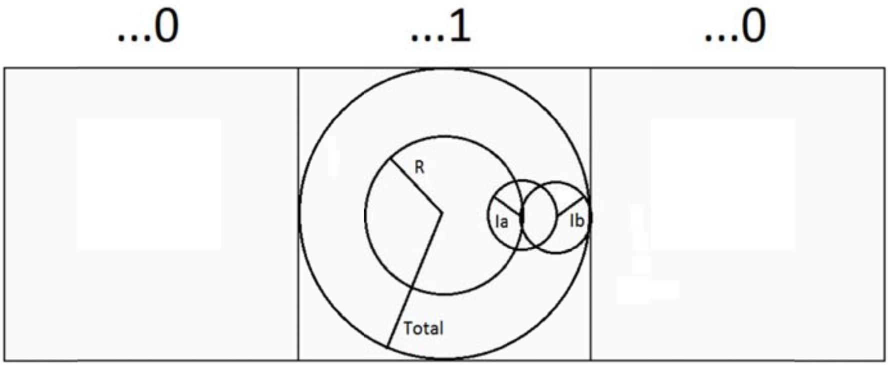

Figure 5-3: Relation between LSB and inaccuracy

Possible grid cell sizes given with the choice of 1 degree per latitude segment (and therefore possible for the rest of Table 5-2 as well) are listed in Table 5-3 with the assumption that $I_a = I_b$. A grid size of 550 meters was chosen to fit 200 cells per degree, 1100 for 100, and 2200 for 50 grid cells, assuming that the distance between the pole and equator is approximately 10000 km.

Table 5-3: Tolerable inaccuracy given a certain number of LSB

Number of LSBs	550 cells	1100 cells	2200 cells
1	-	25	300
2	163	575	1400
3	713	1675	3600
4	1813	3875	8000
5	4013	8275	16800

The source of the location information can then be chosen based upon the requirements of the selected grid size and available number of LSBs. This approach is possible with non-aligned rectangles as well, but will only by described in detail in this section for the aligned rectangles.

The grid size used and the number of LSBs included could either be communicated in the discovery message or it could be given by the ProSe Function.

5.3.3.2 *Non-aligned rectangles*

To encode the location as a cell in a grid of non-aligned rectangles has the benefit of easily adapting to changes in latitude and avoiding the problems that arise with the curvature of the earth. Figure 5-4 shows a group of non-aligned rectangles for which the width is twice times the ProSe range and the height equal to the ProSe range. The sides are labeled with possible encodings for the latitude and longitude. The receiver is marked with a small dot and a circle around it representing the ProSe range.

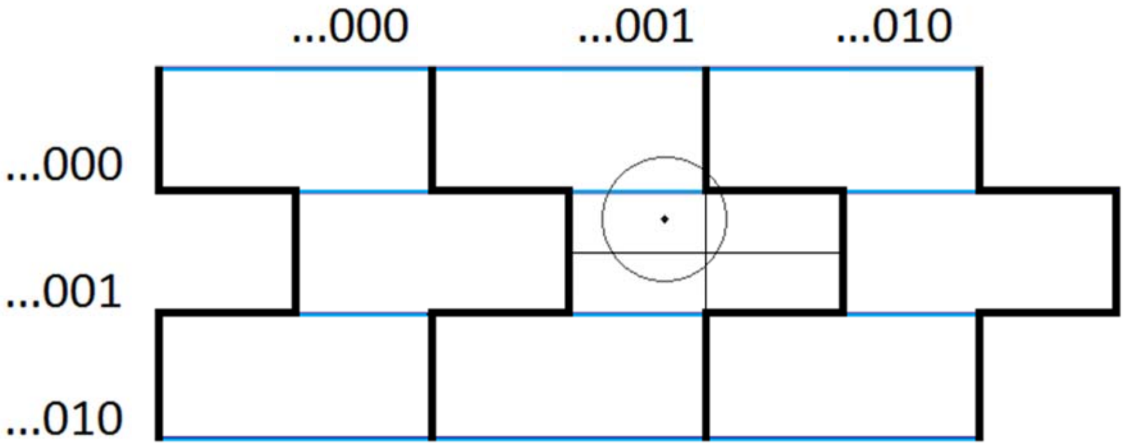

Figure 5-4: A group of non-aligned rectangles

The combinations of rectangles offset from each other and the rectangle's size reduces the number of times the MAC has to be checked in a worst case scenario. Compared to the aligned rectangles' four times worst case, with this type of grid there is one less MAC computation. To reduce the number of times the MAC has to be checked plain text information can be added as with the aligned rectangles. Two bits can encode the specific cell if combined with the receiver's knowledge about its own location within the smaller rectangles in a cell, since for every smaller rectangle there are only 3 possible neighbors. The least significant bit of the latitude and longitude could, as in the case with aligned rectangles, be used to encode this information.

The bits included in the MAC can be chosen using the same method as with aligned rectangles. However, larger cells are needed to achieve a better worst case scenario. To deal with the offset, one can start with the latitude and then add the offset to the longitude depending upon the parity of the encoded latitude. The longitude can then be encoded as well.

5.3.3.3 *Hexagons*

A grid based on hexagons does not have straight lines in any direction (unlike rectangles) and as shown in Figure 5-5 they can be oriented in two different ways. Mapping coordinates to the grid is not as easy as with rectangles, but as with the non-aligned rectangles, the worst case scenario can be reduced to three calculations of the MAC with sufficiently large sides to the hexagons. With a side of twice the ProSe range this worst case scenario is achieved. The receiver's location is marked with a small dot and a circle around representing the ProSe range. The receiver can determine which hexagons to test based on its location within the smaller triangles drawn in the middle hexagon. The two closest hexagons to the triangle the receiver is in are the only ones that need to be tested. In this case the sender can also include information in plain text and the least significant bit of the latitude and longitude is sufficient. This approach is valid for both orientations of the hexagons.

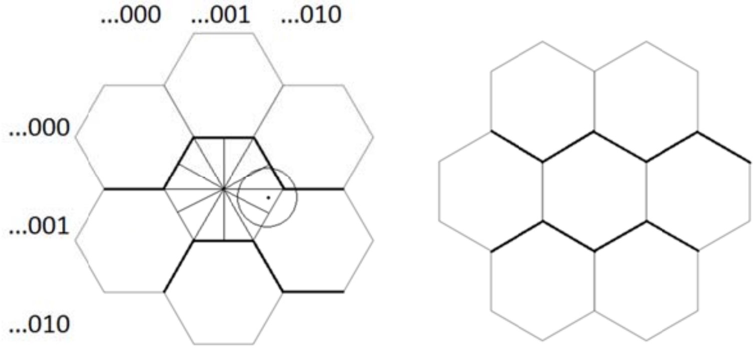

Figure 5-5: Hexagon-based grid

To handle the problem with the curvature of the earth, more effort is needed then for rectangles. One alternative could be as shown in Figure 5-6. In the first case with the orientation of the hexagons as shown for the hexagons in the left of the figure, the last full line of hexagons before the border can be placed at least a ProSe range from it. The hexagons grid can wrap around at 180 degrees East and West forming a continuous belt of hexagons. The hexagon size then has to be chosen such that the belt fits an integer number of them.

In the second case with the orientation of the hexagons as shown in the right of the figure, the two differently sized hexagons can share a common line. The hexagons half a hexagon side from the line are then out of ProSe range. In this case the hexagons can also wrap around at 180 degrees East and West.

The grid cell identifier could either be local to the 'belt' of hexagons of one size, or a total including the half hexagons beside the borders. Both these alternatives solve the curvature problem but will give slightly worse overall performance than a pure hexagon grid, but this additional complexity is needed to avoid too big a difference between the hexagons' area close to the equator and those farther away.

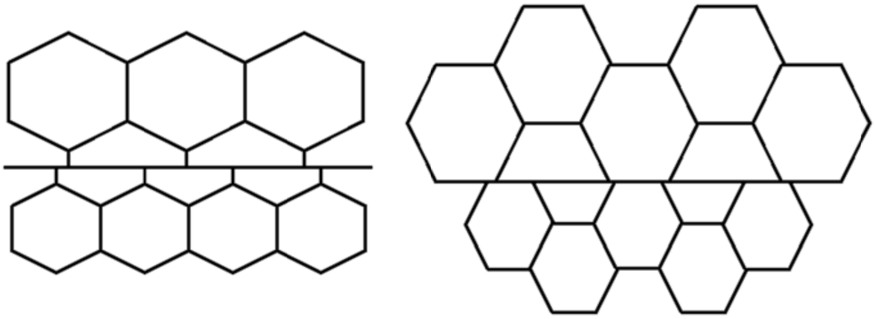

Figure 5-6: Border region of hexagon-based grid

5.3.3.4 *HEALPix*

HEALPix is a class of projections in which every cell, called a pixel, has an equal size, but not equal shape. Every cell can be further divided into 4 equal area pixels. Compared to the previously mentioned alternatives, this grid is already defined and implementations are available.

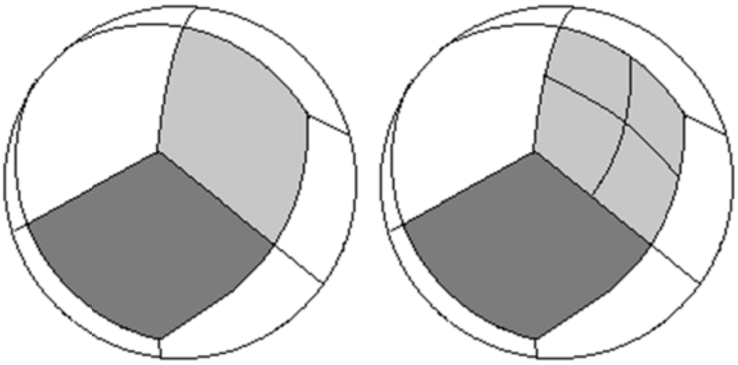

Figure 5-7: HEALPix projection

The area of each cell can easily be retrieved by dividing the earth's surface area by the number of pixels. The possible areas achieved for different numbers of pixels are shown in Table 5-4.

Table 5-4: HEALPix grid cell areas

N	Number of pixels ($12*(2^N)^2$)	Grid cell area
11	50331648	10.13 km^2
12	201326592	2.53 km^2
13	805306368	0.63 km^2

In Table 5-4 and Figure 5-7, the number of 'base areas' are 12 and that is the most commonly used value. Selecting a fixed value as number of 'base areas' results in only a few choices for possible pixel size.

Every pixel has either 7 or 8 neighbors. The worst case scenario for this grid can be reduced to 3 neighboring cells, if the pixels are of sufficient size; therefore, the MAC computation has to be performed up to 4 times in a worst case scenario. Due to the shape of the areas, determining which neighbors that need to be checked is non-trivial, unlike the case with rectangles or hexagons. An approximation may be the most efficient method of doing this. Dividing into 4 times as many areas as needed and determining the neighboring cells based on which smaller area the location represents could be one such approximation.

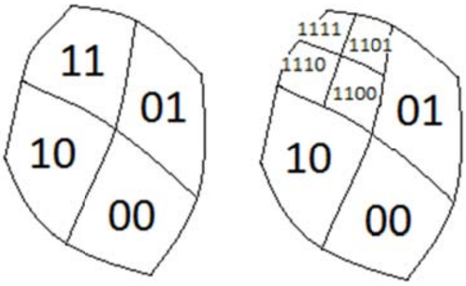

Figure 5-8: HEALPix cell numbering

There are two methods of identifying a certain cell in the grid [72], either nested or ring coordinates. Nested coordinates number cells as in Figure 5-8. Neighboring cells can be distinguished from each other with only the last 2 bits in most places except on e.g. the border of the 3 upper 'base areas', where at least 3 bits is necessary. To solve it, 'base areas' can be given an identifying bit that changes for every area at the same latitude. 3 bits will then be enough to uniquely identify a bordering cell. The same is also the case for the 3 lower, but since there is no land there, the problem will not be as big. Ring coordinates are not as suitable and will therefore not be described here. HEALPix would also potentially infer a lot of problems with floating point precision and consistency between platforms.

5.3.3.5 *Hierarchical Triangular Mesh*

The Hierarchical Triangular Mesh (HTM) divides the earth into triangles of similar shape and size. It is done recursively and in each step, a finer mesh is given (as shown in Figure 5-9. Starting out from 8 base triangles and recursively splitting each into 4 new triangles gives a mesh with fixed sizes.

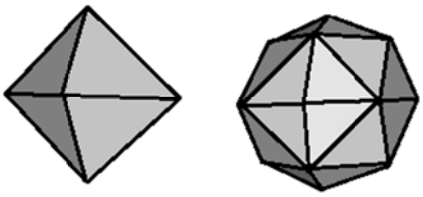

Figure 5-9: Hierarchical Triangular Mesh

The approximate area of each cell can easily be calculated by dividing the earth's surface area by the number of cells. The possible areas achieved for different number of pixels are shown in Table 5-5. HTM is similar to HEALPix in terms of the possible number of areas for each grid cell, but without the possibility of different number of 'base areas'. A grid cell can have up to 12 neighbors and the worst case scenario can be reduced to computing the MAC 6 times given a sufficiently large area.

Table 5-5: HTM grid cell areas

N	Number of pixels $(8*(2^N)^2)$	Grid cell area
11	33554432	15.20 km^2
12	134217728	3.80 km^2
13	536870912	0.95 km^2

Peter Kunszt et al. measured the performance of HTM in [73]. On an old 600 MHz Pentium III they achieved over 50,000 lookups per seconds for level 5 HTM and over 11,000 lookups per second for level 25 HTM. The suitable grid cell areas in Table 5-5 would then probably perform somewhere in between these values.

5.3.3.6 *Military Grid Reference System (MGRS)*

The Military grid reference system (MGRS) is based on Universal Transverse Mercator grid system and the Universal Polar Stereographic grid system. It divides the earth into parts identified by a combination of letters and numbers. An example of an identifier would be 34WED3. The first part is a Grid Zone Designator and then Grid Square ID followed by coordinates to further refine the location identifier. However, the shapes and sizes of the parts vary a lot and some of the cell sizes have been adjusted due to geographical factors. Basing a grid upon MGRS could require a lot of decisions about how irregular rectangles should be divided.

5.3.3.7 *Mixed approach with obfuscated bits*

By including plain text bits to reduce the number of MAC calculations to 1, location privacy is reduced to the same as when the same location identifier is included in the MAC in plain text. To counter this, the bits can be obfuscated. This can be done similarly to how it is done in Restricted Discovery (see Sections 2.2.6.2 and 2.2.6.3), but since so few bits need to be obfuscated, a new calculation is not needed for each message. Processing power can instead be traded for memory. If the result of the HMAC-sha256 calculation is 256 bits, then the input can use the UTC-based counter with the 8 least significant bits set to 0 and then let the 7 (assuming 2 bits are needed for obfuscation) most significant of the 8 least significant of the UTC-based counter determine which pseudo-random bits to be used for the obfuscation. If an announce message is supposed to be sent every 16 seconds and since the pseudo-random bits are enough for 128 messages, a new calculation is only needed every half an hour. A keyed hash function with less output bits could also be used to trade off processing power for memory or only a fraction of the bits could be used.

This mixed approach would only be beneficial in Open Discovery for which the monitoring UE does not have the Discovery Key and in Restricted Discovery when the Discoverer UE is not supplied with the DUIK. Otherwise, the UE would only be protection against other UEs. The solution needs to specify when MAX_OFFSET has been specified as well to know how many bits that needs to be truncated to get the same counter value at both receiver and sender. It is not clear by who, how, and when MAX_OFFSET will be defined.

6 Analysis

Solutions based on monitoring of PC5 with radio fingerprinting will not be analyzed or evaluated because they were determined to be infeasible in practice (at this point in time). However, radio finger printing could be used to supplement other solutions in order to provide increased protection against spatial replay. Checking for multiple uses of announced codes is evaluated in Section 6.1 along with the solutions presented in Section 5.1. Solutions from Section 5.3 are compared in Section 6.2.

Location privacy is evaluated according to Section 3.3. The reference of location privacy with the proximity constraint from the receiving UE becomes as in equation 6-1, where R, the ProSe range, is normalized to 1

$$\frac{1}{R^2\pi}\int_0^R 2\pi R dR = 1 \qquad 6\text{-}1$$

6.1 Network-based approaches

A network based approach has the benefit of not revealing one's location to other UEs. It also separates the other UEs proximity constraint (given by the radio transmitter range) from the location identifier. Together the location privacy is reduced more by a location identifier and the proximity constraint than separately. This difference can be seen when comparing Table 6-2 to Figure 6-2.

However, all of the solutions have some negative aspects, mainly the increased network traffic. The solution based on permissions would increase the necessary number of times for a permission to be acquired and is also limited by the cell's size. A finer grained permission is of course possible, but this would increase the frequency of when a moving UE reports its location and when it must again request permission, therefore this approach will not be considered further. Additionally, because permissions are handled by the HSS there would be added traffic to and from both the UEs and the ProSe Function to the HSS. (Hence the suggestion in Section 5.1.2 for a new reference point between the ProSe Function and the MME.)

SLP could be a more suitable node to involve than the HSS. The SLP is designed to handle location information and message structures could be reused from EPC-level Discovery. Equation 6-2 gives the maximum replay distance d. With R as the ProSe range, v_a and v_b as the velocities of the respective UE, t_s the time the location is reported, t_r the time the distance is estimated, Δ the error in time synchronization between the clocks, and I_a and I_b as the inaccuracies of the location information given by UE A and UE B. Table 6-1 shows the resulting distance for some values of the time difference between sending and receiving. If the location updates are time stamped by the SLP

$$d \leq R + (v_a + v_b)(t_s - t_r + \Delta) + I_a + I_b \qquad 6\text{-}2$$

and compared to the time of receiving the Match Report, Δ becomes close to 0. If the sending UE updates its position every time it sends a Discovery message, the T value should maximally be close to the validity time of a message (16 seconds).

Table 6-1: Table of update frequency and resulting maximum distance

T (seconds)	Distance (meters)
10	1300
15	1600
20	1900

Equation 6-3 gives the quantified location privacy according to Section 3.3 for a square with the side L.

$$\frac{2}{L^2}\int_0^L\int_0^x\sqrt{x^2+y^2}dydx \qquad 6\text{-}3$$

A solution based on already existing tracking areas would not introduce so much extra network traffic. Extra traffic would only be generated when the ProSe Function is checking a MAC. Since tracking areas consists of multiple cells, the protection against spatial replay will also be limited to the aggregate of these cells' coverage area. An advantage is that tracking areas already exist and no extra location information needs to be shared. These solutions are summarized in Table 6-2.

Table 6-2: **Summary of network based solutions**

Solution	Added overhead	Location privacy		Replay distance
		UEs	carriers	
Permission	2 messages every serving cell change	1	-[*]	0 to 100 km (Based upon cell size)
Tracking areas	2 messages every Discovery Report	1	-	0 to >100 km (Size of the several cells)
SLP	1 message every N seconds for sending UEs and 1 message to retrieve location from the SLP	1	0 to 1.91[†]	0 to 500[‡]
Match Report	A few bits of location in the Match Report	1	0 to 1.91	0 to 500

A network based solution will naturally require network coverage for such a solution to work, hence when there is no coverage such a solution will be unavailable. Since coverage is not always the case in Public Safety use, another solution would then be needed for these specific discoveries or they would be left unprotected from spatial replay. There have also been discussions within the 3GPP of extending the coverage less use of ProSe in the future, which would further reduce the applicable of these solutions.

[*] Not affected

[†] Calculated for a precision of 500 meters in the location reported to the network

[‡] Can be chosen arbitrarily, but since the carrier already knows the UEs coarse grained location, there is no point of having worse than about 500

6.2 Discovery message based

An implicit solution would result in the least changes to the standard. The receiving UE would only need to include its location in the Match Report for Open Discovery and if provisioned with the DUIK in Restricted Discovery, no changed would be needed at all.

However, an implicit solution would require more than 1 calculations of the MAC in some cases. Figure 6-1 presents the expected number of calculations needed. The estimated values for the number of calculations have been done as described in the appendices. Both double integrals and Monte Carlo methods have been used to calculate these estimated values. Some of these calculations have been done with both methods to ensure correctness of the respective method. These calculations assume close to perfect accuracy in the location information, but this can easily be adjusted for reality by sliding the curves to the right. The smallest considered grid size is a square with 1 km sides which requires approximately 1.55 times as many required MAC checks as the number of MAC checks as with perfect location accuracy.

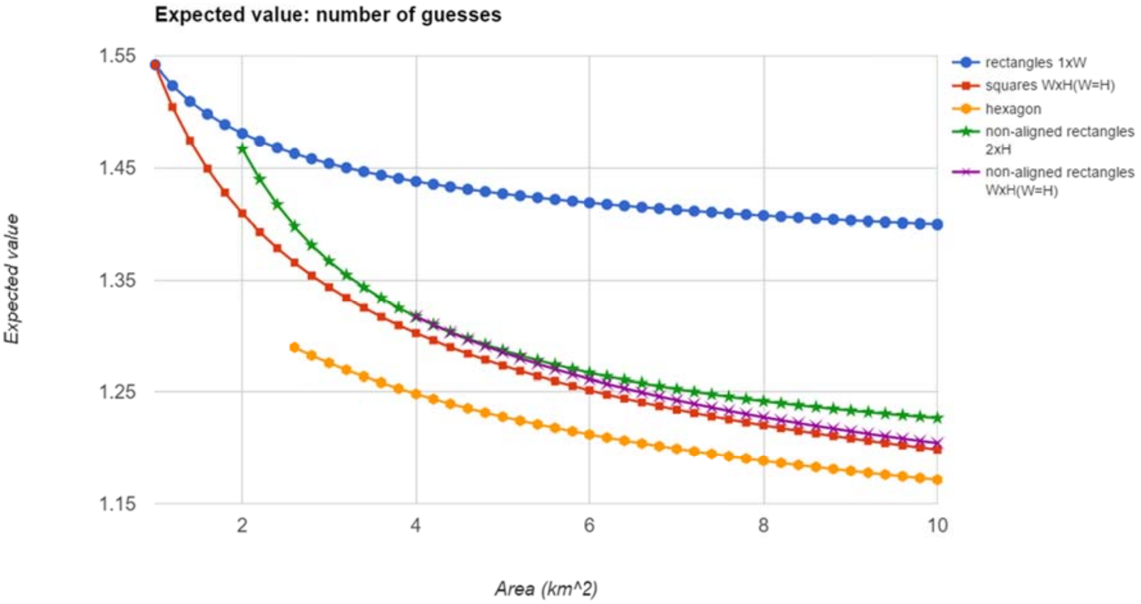

Figure 6-1: Comparison of expected value for the number MAC calculations necessary

No calculations have been done for HEALPix, HTM, or MGRS; but both HEALPix and HTM can be expected to perform worse than the alternatives shown in Figure 6-1 for a given area due to the sharper angles in some of the corners. The values in Figure 6-1 were calculated without shape deformations that will be present close to the poles for some of the grids, but this effect is assumed to be negligible for highly populated latitudes. As HTM is based on triangles its will probably have the worst result since the irregular shape makes it hard to choose the minimum number of neighbors for all the cases. The number of computation needed to map coordinates to unique triangle identifiers with a suitable size has not been determined in this thesis due as no suitable existing code was found for this calculation, but it can be assumed to be not more efficient than the more basic grid types. The results of the benchmark presented in Section 5.3.3.5 are quite old, but suggest that the calculation is relatively demanding. The other values were evaluated on the same modern Intel processor in order to make them comparable to each other. The results of these calculations along with other information are presented in Table 6-3.

Table 6-3: Grid comparison

Grid type	Adaptability	Mixed bits (at least number)	Neighbors (max)	Equal area	Clock cycles for encoding (sender)
HEALPix	Low	Yes(3)	3(8)	Yes	300
HTM	Low	No	6(12)	Close	-
MGRS	Medium	Yes(2)	3(8)	No	-
Aligned Rectangles	Medium	Yes(2)	3(8)	No	10
Aligned Rectangles 2	High	Yes(2)	3(8)	Close	30
Non-aligned Rectangles	Medium	Yes(2)	2(6)	No	20
Hexagons	Low	Yes(2)	2(6)	No	80

HEALPix with other than 12 base areas would perform about the same as the two alternatives shown in the figure and the table, but no implementation was found supporting this calculation. Note that HEALPix requires a lot more computations than the more basic grid type and finding the right neighbors, to minimize calculations of the MAC, requires even more. HEALPix's equal size property is not that important since the alternatives handle the problem with sufficient accuracy results, especially HTM and Aligned Rectangles 2. Additionally, since the choice of sizes are limited, only multiples of 4, being able to choose close to the optimal size may be impossible. HEALPix also requires 1 more bit then the others to support a mixed solution as would be needed if (unnecessary) MAC calculations are to be avoided.

MGRS cells are shaped like rectangles, therefore the result will be the same as shown in the figure for the same shape and size rectangles, but the more complicated cell identifier and how it is calculated gives this alternative unnecessary additional overhead.

As can be seen in Figure 6-2, the different location obfuscation techniques perform very similarly when considered as an average between all possible situations. However, when certain situations are considered, there are some bigger differences. The truncation based technique give away a lot of location privacy when the square given by the location identifier has a small intersection with the another UE's ProSe range. For example, this occurs when the UE is close to a corner. The same is not true for randomized obfuscation; but does occur for a stationary UE, as the average between shared locations will move towards the real location. Since truncation is the technique naturally implemented when mapping locations to a fixed grid, this is the technique used for most of the proposed solutions.

In Open Discovery coordinates can either be in plain text (hence readable by anyone) or be used only in an implicit solution. If included in plain text, then the UE can determine if the message's origin is local or not and then decide to send a Match Report to the ProSe Function to check the message's MAC. If an implicit solution is used, then the UE always need to send a Match Report to the ProSe Function. Either the overhead of the extra location bits will be transported over PC5 or over PC3, since the ProSe function needs the location to check the MAC. The number of messages sent over each interface will be the number sent over PC5 which will be greater than or equal to the number sent over PC3 for a single user. However, with many users the number of message will depend on the usage patterns (how many users that receive and process the same broadcast and how many broadcasts that goes by without interest). A mixed solution would give the same amount of location privacy as an explicit solution in both Restricted Discovery and Open Discovery, while an implicit solution would only reduce the location privacy in Restricted Discovery when the receiving UEs have been provisioned with the DUIK. Location privacy would be protected from non-authorized users without access to the key though.

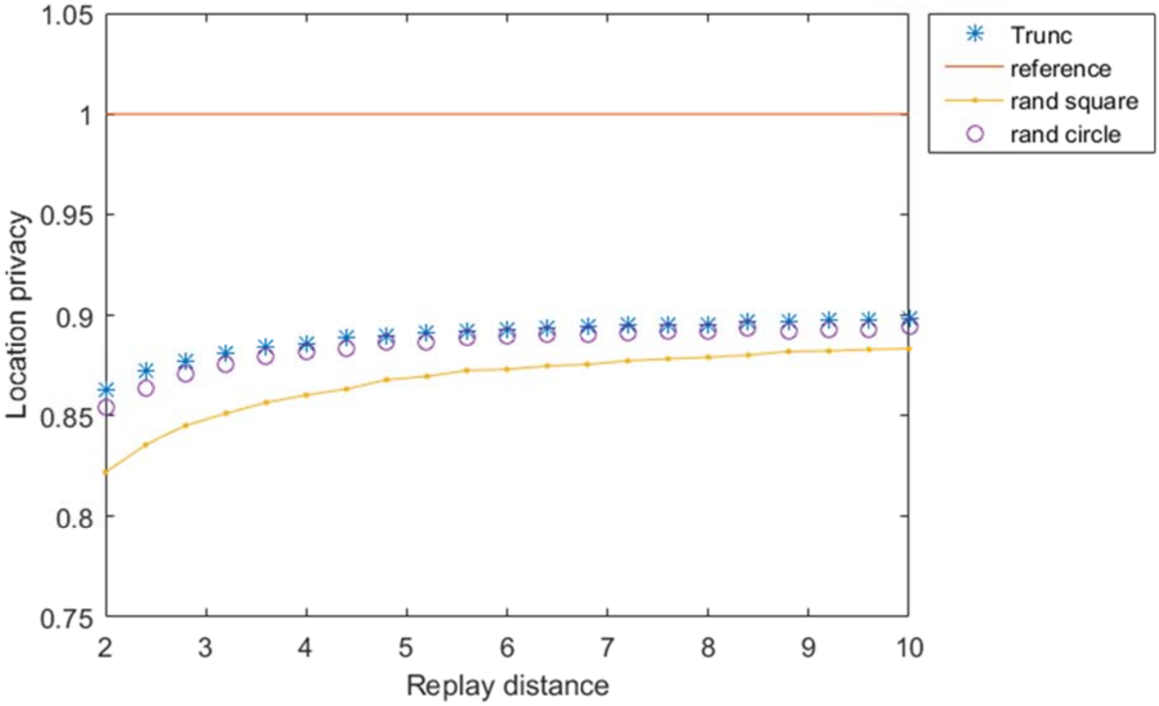

Figure 6-2: **Location privacy for different location obfuscation techniques**

7 Conclusions and Future work

In this chapter the conclusions from Section 6 are presented. Some of the limitations of this work are presented in Section 7.2. Suggestions for future work are presented in Section 7.3. The chapter concludes with some reflections in Section 7.4.

7.1 Conclusions

If the chosen solution needs to support the ProSe functionality when there is no cellular network coverage, then the choices of solutions are drastically reduced. In this project there was a desire for a future proof solution, as a result this leaves only the Discovery message based solutions, specifically the choices are: temporal packet leash, explicit geographical packet leash, and mixed geographical packet leash with the grid type aligned rectangles 2. Unfortunately, an implicit solution would introduce too much potential overhead for the ProSe Function when checking the MAC.

The temporal packet leash would require too many changes in the draft standard to be feasible. Examples of such changes are tightening the timing constraint of sending and the continuous use of GPS. The explicit packet leash introduces less overhead when there are spatial replay attacks present, but otherwise a mixed solution is better.

The solution that has been determined to be most suitable is a mixed solution with the grid type aligned rectangles 2 (see Section 5.3.3.1). This solution can be implemented with or without obfuscation of the location information added to the Discovery Message. If obfuscation is not used, then the suggested grid sizes of 550, 1100, and 2200 meters will preserve 0.69 %, 0.76 %, and 0.87 % of the original location privacy. The chosen grid size can be communicated in the Discovery Message and the ProSe Function can decide upon the balance between location privacy and security to be used by the sending UE. A smaller grid with a certain number of LSBs included will require higher location accuracy by both the sending and receiving UEs.

In contrast, if no location privacy is required or wanted, then stationary UEs should include their exact location in their dynamic metadata (see Section 5.1.4).

7.2 Limitations

A limitation of this thesis is the absence of real users, implementations, and use cases. Since the draft standard is still in the process of being standardized, one cannot reliably predict or test how a solution would actually perform or if a solution is even needed. Therefore, an adaptable solution has been chosen to hopefully cover future needs. Moreover, should it be determined that such a solution is unnecessary, the existence of this solution should impose minimal overhead.

Another limitation is that only solutions based on what is currently available in a modern smartphone is considered (see Section 4.4) and only solutions that can be specified without a lot of real world testing.

7.3 Future work

A clear element of future work is to evaluate the actual need for spatial replay prevention in ProSe and based upon these need reconsider the alternatives. For example, some of the methods that have been ruled out in this thesis, for not being sufficiently mature or is dependent upon hardware that is not common in today's smartphones, could be part of future work.

Future work could also consist of security and privacy aspects of other parts of Proximity Services, such as using coordinate transformations to protect the users' location privacy from other carriers in EPC-level Discovery.

7.4 Reflections

The ProSe standard has the potential to more efficiently utilize spectrum and slightly reduce energy consumption. ProSe also improves communication for Public Safety workers. For a standard to be secure is essential that it preserves the desired security properties (in this case reducing or preventing spatial replay attacks) *and* that it delivers the necessary quality of user experience. This thesis has presented solutions that could potentially reduce the possibilities of misuse of the standard. Unfortunately, several of the solutions, particularly the chosen solution, does reduce the user's location privacy. This reduction of privacy has been quantified and several options are given to balance privacy and security. Such a compromise could have been avoided, but it was determined that it was infeasible to change the standard is as would be necessary to achieve this. Security was therefore prioritized over preserving the maximum possible location privacy. Fortunately, the reduction in location privacy was found to be sufficiently low that it has been determined by my industrial adviser to be acceptable.

The suggestions in this thesis do not affect any other aspects of the standard and as a consequence to not directly raise any ethical or sustainable issues. Moreover, the added communication overhead is small compared to the total amount of data expected to be transferred by the UEs.

References

[1] GNSS Market Report, Issue 4 (2015), 'GNSS Market Report'. [Online]. Available: http://www.gsa.europa.eu/sites/default/files/LBS_0.pdf. [Accessed: 24-Mar-2016]

[2] 3GPP, 'Study on LTE support for Vehicle to Everything (V2X) services (3GPP specification: 22.885)', Dec-2015. [Online]. Available: http://www.3gpp.org/DynaReport/22885.htm. [Accessed: 05-Apr-2016]

[3] 3GPP, 'Functional stage 2 description of Location Services (LCS) (3GPP specification: 23.271)', Sep-2015. [Online]. Available: http://www.3gpp.org/dynareport/23271.htm. [Accessed: 22-Mar-2016]

[4] Jochen Schiller and Agnès Voisard, *Location-Based Services*. Elsevier, 2004, ISBN: 978-0-08-049172-1.

[5] 3GPP, 'Location Services (LCS); Service description; Stage 1 (3GPP specification: 22.071)', Sep-2015. [Online]. Available: http://www.3gpp.org/DynaReport/22071.htm. [Accessed: 22-Mar-2016]

[6] Axel Küpper, 'Introduction', in *Location-Based Services*, John Wiley & Sons, Ltd, 2005, pp. 1–14 [Online]. Available: http://dx.doi.org/10.1002/0470092335.ch1

[7] FCC, 'FCC Acts to Promote Competition and Public Safety in Enhanced Wireless 911 Services'. [Online]. Available: https://transition.fcc.gov/Bureaus/Wireless/News_Releases/1999/nrwl9040.html. [Accessed: 05-May-2016]

[8] FCC, 'FCC Adopts Rules to Implement Enhanced 911 for Wireless Services'. [Online]. Available: https://transition.fcc.gov/Bureaus/Wireless/News_Releases/1996/nrwl6026.txt. [Accessed: 05-May-2016]

[9] Jie Liu, Bodhi Priyantha, Ted Hart, Heitor S. Ramos, Antonio A. F. Loureiro, and Qiang Wang, 'Energy Efficient GPS Sensing with Cloud Offloading', in *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, New York, NY, USA, 2012, pp. 85–98 [Online]. DOI: 10.1145/2426656.2426666

[10] Jeongyeup Paek, Joongheon Kim, and Ramesh Govindan, 'Energy-efficient Rate-adaptive GPS-based Positioning for Smartphones', in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, New York, NY, USA, 2010, pp. 299–314 [Online]. DOI: 10.1145/1814433.1814463

[11] Kaisen Lin, Aman Kansal, Dimitrios Lymberopoulos, and Feng Zhao, 'Energy-accuracy Trade-off for Continuous Mobile Device Location', in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, New York, NY, USA, 2010, pp. 285–298 [Online]. DOI: 10.1145/1814433.1814462

[12] Google, 'The Google Maps Geolocation API', *Google Developers*. [Online]. Available: https://developers.google.com/maps/documentation/geolocation/intro. [Accessed: 04-Jul-2016]

[13] Intel, 'POLS'. [Online]. Available: http://pols.sourceforge.net/. [Accessed: 13-Apr-2016]

[14] 'OpenCellID - OpenCellID'. [Online]. Available: http://opencellid.org/. [Accessed: 04-Jul-2016]

[15] Carlos H. Aldana, 'Methods and systems for positioning based on observed difference of time of arrival', US Patent Number: US9075125 B2, 07-Jul-2015 [Online]. Available: http://www.google.com.gt/patents/US9075125. [Accessed: 15-Jul-2016]

[16] Andreas Waadt, Guido H. Bruck, Peter Jung, João Figueiras, and Simone Frattasi, 'Positioning Systems and Technologies', in *Mobile Positioning and Tracking*, John Wiley & Sons, Ltd, 2010, pp. 177–211 [Online]. Available: http://dx.doi.org/10.1002/9780470663035.ch8

[17] 3GPP, 'Universal Geographical Area Description (GAD) (3GPP specification: 23.032)', Dec-2015. [Online]. Available: http://www.3gpp.org/DynaReport/23032.htm. [Accessed: 06-May-2016]

[18] 3GPP, 'Proximity-based services (ProSe); Stage 2 (3GPP specification: 23.303 version 13.2.0 Release 13)', Mar-2016. [Online]. Available: http://www.3gpp.org/DynaReport/23303.htm. [Accessed: 05-Apr-2016]

[19] Qualcomm, 'LTE Direct | Research Project', *Qualcomm*, 29-Apr-2014. [Online]. Available: https://www.qualcomm.com/invention/research/projects/lte-direct. [Accessed: 07-Jul-2016]

[20] 3GPP, 'Feasibility study for Proximity Services (ProSe) (3GPP specification: 22.803)', Jun-2013. [Online]. Available: http://www.3gpp.org/DynaReport/22803.htm. [Accessed: 02-Apr-2016]

[21] 3GPP, 'Proximity-based Services (ProSe); Security aspects (3GPP specification: 33.303)', Mar-2016. [Online]. Available: http://www.3gpp.org/DynaReport/33303.htm. [Accessed: 06-Apr-2016]

[22] Wi-Fi Alliance, 'How far does a Wi-Fi Direct connection travel?' [Online]. Available: https://www.wi-fi.org/knowledge-center/faq/how-far-does-a-wi-fi-direct-connection-travel. [Accessed: 11-Apr-2016]

[23] *Personuppgiftslag (1998:204)*. 1998 [Online]. Available: https://lagen.nu/1998:204. [Accessed: 13-May-2016]

[24] 'EUR-Lex - 31995L0046 - EN', *Official Journal L 281 , 23/11/1995 P. 0031 - 0050;* [Online]. Available: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31995L0046:en:HTML. [Accessed: 13-May-2016]

[25] National Coordination Office for Space-Based Positioning, Navigation, and Timing, 'GPS.gov: Geolocation Privacy Legislation'. [Online]. Available: http://www.gps.gov/policy/legislation/gps-act/. [Accessed: 13-May-2016]

[26] Jinyan Zang, Krysta Dummit, James Graves, Paul Lisker, and Latanya Sweeney, 'Who Knows What About Me? A Survey of Behind the Scenes Personal Data Sharing to Third Parties by Mobile Apps', *Technol. Sci.*, Oct. 2015 [Online]. Available: http://techscience.org/a/2015103001/. [Accessed: 13-May-2016]

[27] Hazim Almuhimedi, Florian Schaub, Norman Sadeh, Idris Adjerid, Alessandro Acquisti, Joshua Gluck, Lorrie Faith Cranor, and Yuvraj Agarwal, 'Your Location Has Been Shared 5,398 Times!: A Field Study on Mobile App Privacy Nudging', in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, New York, NY, USA, 2015, pp. 787–796 [Online]. DOI: 10.1145/2702123.2702210

[28] George Danezis, Stephen Lewis, and Ross Anderson, 'How much is location privacy worth', in *In Proceedings of the Workshop on the Economics of Information Security Series (WEIS*, 2005.

[29] Dan Cvrcek, Marek Kumpost, Vashek Matyas, and George Danezis, 'A Study on the Value of Location Privacy', in *Proceedings of the 5th ACM Workshop on Privacy in Electronic Society*, New York, NY, USA, 2006, pp. 109–118 [Online]. DOI: 10.1145/1179601.1179621

[30] John Krumm, 'A Survey of Computational Location Privacy', *Pers. Ubiquitous Comput*, vol. 13, no. 6, pp. 391–399, Aug. 2009. DOI: 10.1007/s00779-008-0212-5

[31] Eija Kaasinen, 'User Needs for Location-aware Mobile Services', *Pers. Ubiquitous Comput*, vol. 7, no. 1, pp. 70–79, May 2003. DOI: 10.1007/s00779-002-0214-7

[32] Louise Barkhuus and Anind Dey, 'Location-Based Services for Mobile Telephony: a Study of Users' Privacy Concerns', in *Proceedings of the 9TH IFIP TC13 International Conference on Human-Computer Interaction (INTERACT 2003),* 2003, pp. 709–712.

[33] Janice Y Tsai, Patrick Gage Kelley, Lorrie Faith Cranor, and Norman Sadeh, ‘Location-sharing technologies: Privacy risks and controls’, *ISJLP*, vol. 6, p. 119, 2010.

[34] Marius Wernke, Pavel Skvortsov, Frank Dürr, and Kurt Rothermel, ‘A classification of location privacy attacks and approaches’, *Pers. Ubiquitous Comput.*, vol. 18, no. 1, pp. 163–175, Jan. 2014. DOI: 10.1007/s00779-012-0633-z

[35] C. A. Ardagna, M. Cremonini, E. Damiani, S. De Capitani di Vimercati, and P. Samarati, ‘Location privacy protection through obfuscation-based techniques’, presented at the Proceedings of the 21st annual IFIP WG 11.3 working conference on Data and applications security, 2007, pp. 47–60 [Online]. Available: http://dl.acm.org.focus.lib.kth.se/citation.cfm?id=1770560.1770566. [Accessed: 30-Mar-2016]

[36] Pierangela Samarati and Latanya Sweeney, ‘Protecting Privacy when Disclosing Information: k-Anonymity and Its Enforcement through Generalization and Suppression’, 1998.

[37] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam, ‘L-diversity: Privacy Beyond K-anonymity’, *ACM Trans Knowl Discov Data*, vol. 1, no. 1, Mar. 2007 [Online]. DOI: 10.1145/1217299.1217302

[38] N. Li, T. Li, and S. Venkatasubramanian, ‘t-Closeness: Privacy Beyond k-Anonymity and l-Diversity’, in *2007 IEEE 23rd International Conference on Data Engineering*, 2007, pp. 106–115. DOI: 10.1109/ICDE.2007.367856

[39] Marco Gruteser and Dirk Grunwald, ‘Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking’, in *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services*, New York, NY, USA, 2003, pp. 31–42 [Online]. DOI: 10.1145/1066116.1189037

[40] Chi-Yin Chow, Mohamed F. Mokbel, and Xuan Liu, ‘A Peer-to-peer Spatial Cloaking Algorithm for Anonymous Location-based Service’, in *Proceedings of the 14th Annual ACM International Symposium on Advances in Geographic Information Systems*, New York, NY, USA, 2006, pp. 171–178 [Online]. DOI: 10.1145/1183471.1183500

[41] H. Kido, Y. Yanagisawa, and T. Satoh, ‘An anonymous communication technique using dummies for location-based services’, in *Pervasive Services, 2005. ICPS ’05. Proceedings. International Conference on*, 2005, pp. 88–97. DOI: 10.1109/PERSER.2005.1506394

[42] A. Escudero-Pascual and G. Q. Maguire, ‘Role(s) of a proxy in location based services’, in *Personal, Indoor and Mobile Radio Communications, 2002. The 13th IEEE International Symposium on*, 2002, vol. 3, pp. 1252–1256 vol.3. DOI: 10.1109/PIMRC.2002.1045229

[43] A. R. Beresford and F. Stajano, ‘Location privacy in pervasive computing’, *IEEE Pervasive Comput.*, vol. 2, no. 1, pp. 46–55, Jan. 2003. DOI: 10.1109/MPRV.2003.1186725

[44] A. Gutscher, ‘Coordinate transformation - a solution for the privacy problem of location based services?’, in *Proceedings 20th IEEE International Parallel Distributed Processing Symposium*, 2006, p. 7 pp.-pp. DOI: 10.1109/IPDPS.2006.1639681

[45] R. Shokri, G. Theodorakopoulos, J. Y. Le Boudec, and J. P. Hubaux, ‘Quantifying Location Privacy’, in *2011 IEEE Symposium on Security and Privacy*, 2011, pp. 247–262. DOI: 10.1109/SP.2011.18

[46] 3GPP, ‘Vocabulary for 3GPP Specifications (3GPP specification: 21.905)’, Dec-2015. [Online]. Available: http://www.3gpp.org/DynaReport/21905.htm. [Accessed: 05-Apr-2016]

[47] Mihir Bellare, Ran Canetti, and Hugo Krawczyk, 'Keying Hash Functions for Message Authentication', in *Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology*, London, UK, UK, 1996, pp. 1–15.

[48] Hugo Krawczyk, Ran Canetti, and Mihir Bellare, 'HMAC: Keyed-Hashing for Message Authentication'. [Online]. Available: https://tools.ietf.org/html/rfc2104. [Accessed: 06-Apr-2016]

[49] Christof Paar and Jan Pelzl, 'Hash Functions', in *Understanding Cryptography*, Springer Berlin Heidelberg, 2010, pp. 293–317.

[50] Christof Paar and Jan Pelzl, 'Message Authentication Codes (MACs)', in *Understanding Cryptography: A Textbook for Students and Practitioners*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 319–330 [Online]. Available: http://dx.doi.org/10.1007/978-3-642-04101-3_12

[51] Tuomas Aura, 'Strategies against replay attacks', in *Computer Security Foundations Workshop, 1997. Proceedings., 10th*, 1997, pp. 59–68. DOI: 10.1109/CSFW.1997.596787

[52] Paul Syverson, 'A taxonomy of replay attacks [cryptographic protocols]', in *Computer Security Foundations Workshop VII, 1994. CSFW 7. Proceedings*, 1994, pp. 187–191. DOI: 10.1109/CSFW.1994.315935

[53] Stephen Kent, 'IP Authentication Header'. [Online]. Available: https://tools.ietf.org/html/rfc4302. [Accessed: 15-Jul-2016]

[54] Stephen Kent, 'IP Encapsulating Security Payload (ESP)'. [Online]. Available: https://tools.ietf.org/html/rfc4303. [Accessed: 15-Jul-2016]

[55] David A. McGrew and Karl Norrman, 'The Secure Real-time Transport Protocol (SRTP)'. [Online]. Available: https://tools.ietf.org/html/rfc3711. [Accessed: 15-Jul-2016]

[56] U. Carlsen, 'Cryptographic protocol flaws: know your enemy', in *Computer Security Foundations Workshop VII, 1994. CSFW 7. Proceedings*, 1994, pp. 192–200. DOI: 10.1109/CSFW.1994.315934

[57] Thomas Y. C. Woo and Simon S. Lam, 'A lesson on authentication protocol design', *ACM SIGOPS Oper. Syst. Rev.*, vol. 28, no. 3, pp. 24–37, Jul. 1994. DOI: 10.1145/182110.182113

[58] Streekanth Malladi, Jim Alves-Foss, and Robert B. Heckendorn, 'On Preventing Replay Attacks on Security Protocols', in *In Proc. International Conference on Security and Management*, 2002, pp. 77–83.

[59] Apple, 'Getting Started with iBeacon', Jun-2014. [Online]. Available: https://developer.apple.com/ibeacon/Getting-Started-with-iBeacon.pdf. [Accessed: 12-May-2016]

[60] Tengqingqing Ge, 'Indoor Positioning System based on Bluetooth Low Energy for Blind or Visually Impaired Users : Running on a smartphone', Master's Thesis, KTH Royal Institute of Technology, Stockholm, Sweden, 2015 [Online]. Available: http://kth.diva-portal.org/smash/record.jsf?pid=diva2%3A861838&dswid=-1052. [Accessed: 15-Jul-2016]

[61] ETSI, 'TS 100 392-7 - V2.4.1 - Terrestrial Trunked Radio (TETRA); Voice plus Data (V+D); Part 7: Security', Oct-2006. [Online]. Available: http://www.etsi.org/deliver/etsi_ts/100300_100399/10039207/02.04.01_60/ts_10039207v020401p.pdf. [Accessed: 12-May-2016]

[62] ETSI, 'EN 300 396-6 - V1.5.1 - Terrestrial Trunked Radio (TETRA); Direct Mode Operation (DMO); Part 6: Security', Sep-2012. [Online]. Available: http://www.etsi.org/deliver/etsi_en/300300_300399/30039606/01.05.01_60/en_30039606v010501p.pdf. [Accessed: 12-May-2016]

[63] Y. C. Hu, A. Perrig, and D. B. Johnson, 'Packet leashes: a defense against wormhole attacks in wireless networks', in *INFOCOM 2003. Twenty-Second Annual Joint*

Conference of the IEEE Computer and Communications. IEEE Societies, 2003, vol. 3, pp. 1976–1986 vol.3. DOI: 10.1109/INFCOM.2003.1209219

[64] Reza Shokri, Marcin Poturalski, Gael Ravot, Panos Papadimitratos, and Jean-Pierre Hubaux, ‘A Practical Secure Neighbor Verification Protocol for Wireless Sensor Networks’, in *Proceedings of the Second ACM Conference on Wireless Network Security*, New York, NY, USA, 2009, pp. 193–200 [Online]. DOI: 10.1145/1514274.1514302

[65] Suhas Mathur, Robert Miller, Alexander Varshavsky, Wade Trappe, and Narayan Mandayam, ‘ProxiMate: Proximity-based Secure Pairing Using Ambient Wireless Signals’, in *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services*, New York, NY, USA, 2011, pp. 211–224 [Online]. DOI: 10.1145/1999995.2000016

[66] Sejun Song, Hyungbae Park, and Baek-Young Choi, ‘E-LPG: Energy Efficient Location Privacy Scheme Against Global Attackers in Sensor Networks’, *Int. J. Secur. Its Appl.*, vol. 7, no. 2, pp. 27–46.

[67] Paul Johannesson and Erik Perjons, ‘Introduction’, in *An Introduction to Design Science*, Springer International Publishing, 2014, pp. 1–19 [Online]. Available: http://link.springer.com.focus.lib.kth.se/chapter/10.1007/978-3-319-10632-8_1. [Accessed: 28-Jul-2016]

[68] K. Bonne Rasmussen and S. Capkun, ‘Implications of radio fingerprinting on the security of sensor networks’, in *Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on*, 2007, pp. 331–340. DOI: 10.1109/SECCOM.2007.4550352

[69] Stephen Kent, ‘IEEE SA - 1588-2008 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems’. [Online]. Available: https://standards.ieee.org/findstds/standard/1588-2008.html. [Accessed: 19-Jul-2016]

[70] Elham Khorami, ‘Providing accurate time information to a radio base station via a GPS receiver emulator’, Master’s Thesis, KTH Royal Institute of Technology, Stockholm, Sweden, 2013 [Online]. Available: http://kth.diva-portal.org/smash/record.jsf?pid=diva2%3A604497&dswid=5848. [Accessed: 19-Jul-2016]

[71] Mozhdeh Kamel, ‘Extending the precision time protocol to a metropolitan area network : Synchronizing radio base stations’, Master’s Thesis, KTH Royal Institute of Technology, Stockholm, Sweden, 2014 [Online]. Available: http://kth.diva-portal.org/smash/record.jsf?pid=diva2%3A694913&dswid=9803. [Accessed: 19-Jul-2016]

[72] W. O’Mullane, A. J. Banday, K. M. Górski, P. Kunszt, and A. S. Szalay, ‘Splitting the Sky - HTM and HEALPix’, in *Mining the Sky*, A. J. Banday, S. Zaroubi, and M. Bartelmann, Eds. Springer Berlin Heidelberg, 2001, pp. 638–648 [Online]. Available: http://link.springer.com.focus.lib.kth.se/chapter/10.1007/10849171_84. [Accessed: 05-Jul-2016]

[73] Peter Z. Kunszt, Alexander S. Szalay, and Aniruddha R. Thakar, ‘The Hierarchical Triangular Mesh’, in *Mining the Sky: Proceedings of the MPA/ESO/MPE Workshop Held at Garching, Germany, July 31 - August 4, 2000*, A. J. Banday, S. Zaroubi, and M. Bartelmann, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 631–637 [Online]. Available: http://dx.doi.org/10.1007/10849171_83

Appendix A: Calculations

Expected value for aligned rectangles

The expected value E_G in equation 7-1

$$E_G = \frac{1}{A_{R_{1st}}\pi}\iint_{R_{1st}} A_{R_{1st}} + 2A_{R_{2nd}} + 3A_{R_{3rd}} + 4A_{R_{4th}}\, dA \qquad 7\text{-}1$$

for the number of guesses needed to find the right rectangle. A_R represents the area covered by the ProSe range within the rectangle with the corresponding number and the rectangles are numbered after in which order they would be tested. The edge(E) is a square with each side equal to the ProSe range. The upper(U) area other side is equal to the width(W)-2R, the side(S) area other side is height(H)-2R and that leaves the center(C) area to be W-2R times H-2R. All lengths are specified in multiples of R in the calculations to avoid unnecessary complexity.

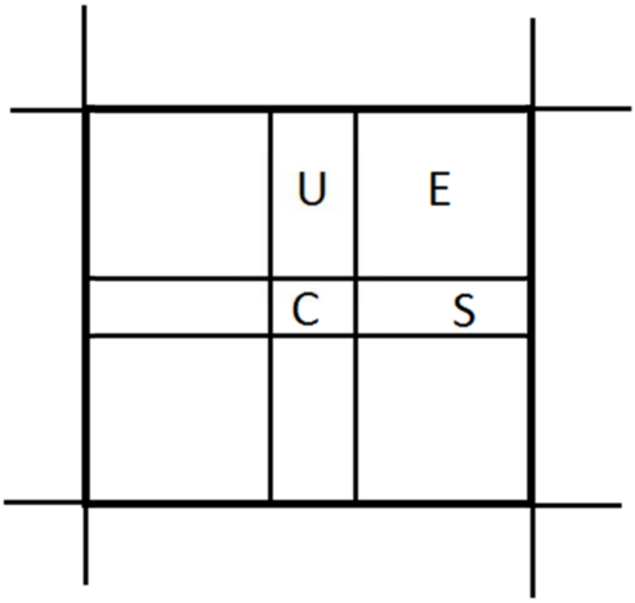

Figure 7-1: Area parts for expected value calculation for aligned rectangles

$$C = \iint_C A_{R_{1st}} dA = \int_0^{W-2}\int_0^{H-2} \pi dydx = \pi(W-2)(H-2) \qquad 7\text{-}2$$

For the center area only R-1st will be covered by R.

For the upper area both the first square and the second square above it will be covered by R. To simplify the formulas, CS_x and CS_y are defined. They represent the area of the smaller part of a circle cut off by a line.

$$CS_x = cos^{-1}(1-x) - (1-x)\sqrt{1^2-(1-x)^2}$$

$$CS_y = cos^{-1}(1-y) - (1-y)\sqrt{1^2-(1-y)^2}$$

$$U = \iint_U A_{R-1st} + 2A_{R_{2nd}}dA = \int_0^{W-2} dx \int_0^1 \pi - CS_y + 2CS_y dy = \int_0^{W-2} dx \int_0^1 \pi + CS_y dy$$

And similarly for the side, but now the second square is the one to the right instead.

$$S = \iint_S A_{R-1st} + 2A_{R_{2nd}}dA = \int_0^{H-2} dy \int_0^1 \pi - CS_x + 2CS_x dx = \int_0^{H-2} dy \int_0^1 \pi + CS_x dx$$

The edge is not as easy to calculate, but will be constant as W and H changes (if kept over 2R as intended). To keep track of which area corresponds to which guess number symmetry can be used. To split the area into two halves like Figure 7-2 in and numbering the squares as in the figure equals the use of the optimal guessing algorithm. The integral then has to be done in three parts. A1 and A2 corresponds to case 1 and A3 corresponds to case 2.

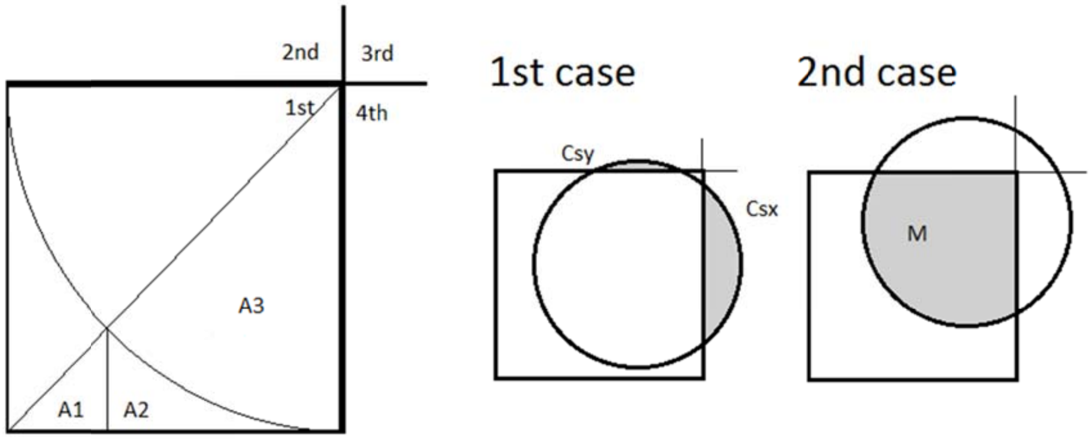

Figure 7-2: **Calculation of the edge using symmetry**

$\mathrm{E} = 2(\mathrm{A1} + \mathrm{A2} + \mathrm{A3})$

For A1:

$$\mathrm{A1} = \iint_{A_1} A_{R_{1st}} + 2A_{R_{2nd}} + 3A_{3rd} dA = \int_0^{1-\frac{1}{sqrt(2)}} \int_0^x G(x,y)\, dydx$$

And for A2:

$$A2 = \iint_{A_2} A_{R_{1st}} + 2A_{R_{2nd}} + 3A_{3rd} dA = \int_{1-\frac{1}{sqrt(2)}}^{1} \int_0^{1-\sqrt{1-(1-x)^2}} G(x,y) dydx$$

In which

$$G(x,y) = \left(\pi - CS_x - CS_y\right) + 2\left(\pi - \left(\pi - CS_x - CS_y\right) - CS_y\right) + 3\left(\pi - \left(\pi - CS_x - CS_y\right) - CS_X\right) = \pi + CS_x + 2CS_y$$

And for A3:

$$A3 = \iint_{A_3} A_{R_{1st}} + 2A_{R_{2nd}} + 3A_{3rd} dA = \iint_{A_3} M + 2\left(\pi - M - CS_y\right) + 3(\pi - M - CS_x) + 4\left(M + CS_x + CS_y - \pi\right) dA = \int_{1-\frac{1}{sqrt(2)}}^{1} \int_{1-\sqrt{1-(1-x)^2}}^{x} \pi + CS_x + 2CS_y \,\mathrm{dydx}$$

In which

$$M = \frac{\pi(2\pi - \frac{\pi}{2} - cos^{-1}(1-x) - cos^{-1}(1-y)}{2pi} + \sqrt{1-(1-x)^2}\frac{1-x}{2} + \sqrt{1-(1-y)^2}\frac{1-y}{2} + (1-x)(1-y)\,,$$

but it does not contribute to the result.

Assuming H>2R and W>2R gives the following formula for the expected value for the number of guesses:

$$E_G \approx \frac{\pi(W-2)(H-2) + 2(H-2)\left(\frac{4}{3}\pi\right) + 2(W-2)\left(\frac{4}{3}\pi\right) + 19.3784}{W * H * \pi} \qquad 7\text{-}3$$

Expected value for non-aligned rectangles

The expected value E_G in Equation 7-4

$$\mathrm{E_G} = \frac{1}{\mathrm{A_{R_{1st}}}\pi} \iint_{\mathrm{R_{1st}}} \mathrm{A_{R_{1st}}} + 2\mathrm{A_{R_{2nd}}} + 3\mathrm{A_{R_{3rd}}}\,\mathrm{dA} \qquad 7\text{-}4$$

for the number of guesses needed to find the right rectangle. A_R represents the area covered by the ProSe range within the a rectangles and the rectangles are numbered after in which order they would be tested. In Figure 7-3 the rectangle has been divided into smaller areas to simplify the calculations. AA and AC are squares with the sides equal to the ProSe range. AD's other side is the height(H)-2 and AB's other side is (width(W)-4)/2. The last area, AE, is then (H-2) times (W-4)/2.

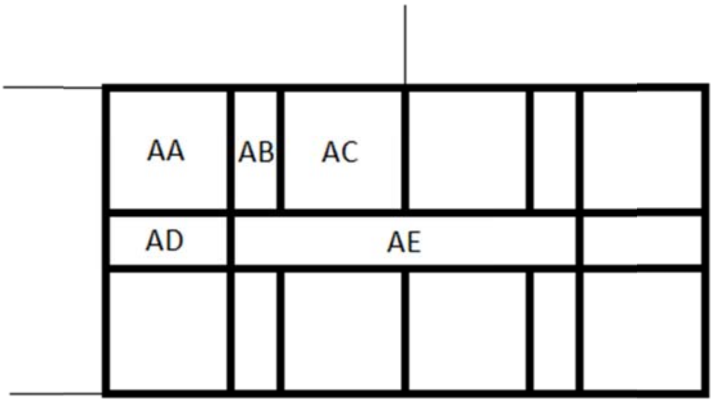

Figure 7-3: **Area parts for expected value calculation for non-aligned rectangles**

$$E_G = 4AA + 4AB + 4AC + 2AD + AE \qquad 7\text{-}5$$

Like for the aligned rectangles Csx and Csy can be defined. Again representing the area of the smaller part of a circle cut off by a line.

$$CS_x = cos^{-1}(1-x) - (1-x)\sqrt{1^2 - (1-x)^2}$$

$$CS_y = cos^{-1}(1-y) - (1-y)\sqrt{1^2 - (1-y)^2}$$

Then AB can is given by:

$$\mathrm{AB} = \iint_{\mathrm{AB}} \mathrm{A_{R-1st}} + 2\mathrm{A_{R_{2nd}}}\mathrm{dA} = \int_0^{\frac{\mathrm{W-4}}{2}} \mathrm{dx} \int_0^1 \pi - CS_y + 2CS_y\mathrm{dy}$$

And AD by:

$$\mathrm{AD} = \iint_{\mathrm{AD}} \mathrm{A_{R-1st}} + 2\mathrm{A_{R_{2nd}}}\mathrm{dA} = \int_0^{\mathrm{H-2}} \mathrm{dy} \int_0^1 \pi - CS_x + 2CS_x\mathrm{dx}$$

For AE only R-1st will be covered by R.

$$AE = \iint_{AE} A_{R-1st} dA = \int_0^{W-2} \int_0^{H-2} \pi dA = \pi(W-2)(H-2)$$

AC is similar to the edge for aligned rectangles, but with only 2 neighbouring rectangles. Again there are two cases to consider, see Figure 7-4.

$\mathrm{AE} = \mathrm{A1} + \mathrm{A2}$

For A1:

$$\mathrm{A1} = \iint_{A_1} A_{R_{1st}} + 2A_{R_{2nd}} + 3A_{3rd}dA = \int_0^{1-\frac{1}{sqrt(2)}} \int_0^x \left(\pi - CS_y\right) + 2CS_y\, dydx = \int_0^{1-\frac{1}{sqrt(2)}} \int_0^x \pi + CS_y\, dydx$$

And for A2:

$$A3 = \iint_{A_3} A_{R_{1st}} + 2A_{R_{2nd}} + 3A_{3rd}dA = \iint_{A_3} \pi - CS_y + 2(\pi - M - CS_x) + 3\left(M + CS_x + CS_y - \pi\right)dA =$$
$$\int_{1-\frac{1}{sqrt(2)}}^{1} \int_{1-\sqrt{1-(1-x)^2}}^{x} M + 3CS_x + 2CS_y \text{ dydx}$$

In which

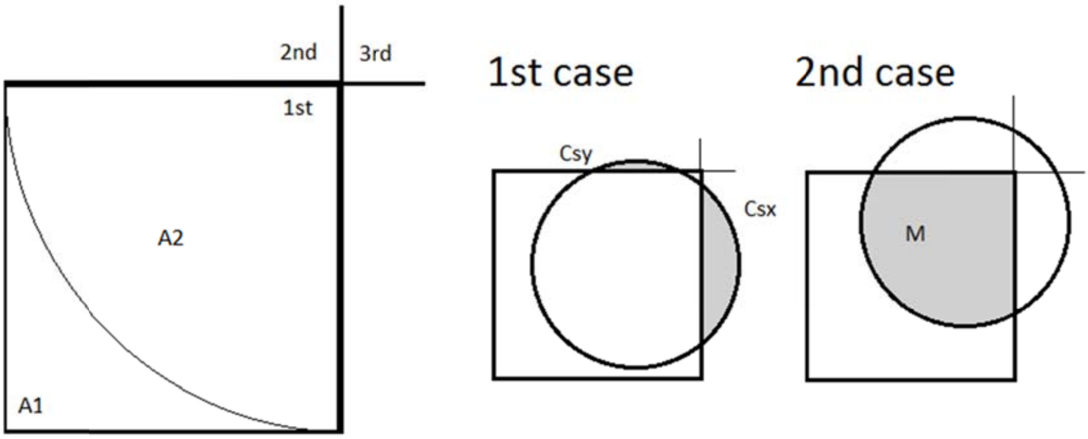

Figure 7-4: **Calculation of AE using symmetry**

$$M = \frac{\pi(2\pi - \frac{\pi}{2} - cos^{-1}(1-x) - cos^{-1}(1-y)}{(2\pi)} + \sqrt{1-(1-x)^2}\frac{1-x}{2} + \sqrt{1-(1-y)^2}\frac{1-y}{2} + (1-x)(1-y)\,,$$

The last area AA is not as easy to do analytically. AA was therefor only done with Monte Carlo method, see Appendix B.

The resulting formula is:

$$E_G = \frac{4(1.557\pi) + 4\frac{(\mathrm{W}-4)}{2}(4/3\pi) + 44.3259 + 2(\mathrm{H}-2)(4/3\pi) + \pi(\mathrm{H}-2)(\mathrm{W}-2)}{WH\pi} \qquad 7\text{-}6$$

Expected value for hexagons

The expected value is given in equation 7-7 like for all the other shapes. Hexagons could have been done completely analytically, but a mixed approach was chosen.

$$\mathrm{E_G} = \frac{1}{\mathrm{R_{1st}}\pi}\iint_{\mathrm{R_{1st}}} \mathrm{A_{R_{1st}}} + 2\mathrm{A_{R_{2nd}}} + 3\mathrm{A_{R_{3rd}}}\,\mathrm{dA} \qquad 7\text{-}7$$

The different area parts for a hexagon can be seen in Figure 7-5. All C areas combined can be thought of as a hexagon with the side equal to twice the ProSe range and H as a hexagon with the side equal to the large hexagons side(W) minus twice the ProSe range. The other side of S is equal to the ProSe range and E’s other side is $\frac{1}{\tan(30)} - 1$.

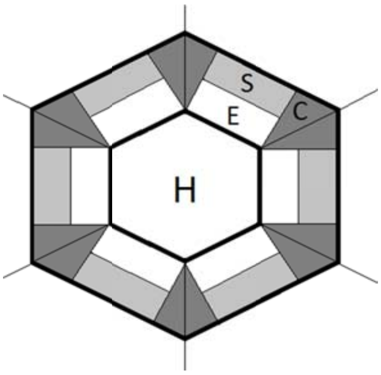

Figure 7-5: **Area parts for expected value calculation for hexagons**

C was estimated using Monte Carlo method, see Appendix B.

$$\mathrm{C} = \frac{13.4\pi}{12}$$

S can be calculated by viewing the hexagon rotated 30 degrees counter-clockwise and utilizing CSy from previous calculations.

$$\mathrm{S} = \iint_{A_S} \mathrm{A}_{\mathrm{H_{1st}}} + 2\mathrm{A}_{\mathrm{H_{2nd}}} \mathrm{dA} = \int_0^{W-2} \mathrm{dx} \int_0^1 \pi - CS_y + 2CS_y \mathrm{dy} = (W-2)(4/3\pi)$$

$$CS_y = cos^{-1}(1-y) - (1-y)\sqrt{1^2 - (1-y)^2}$$

For H, only the first hexagon gives a contribution.

$$H = \pi \frac{3\sqrt{3}}{2}(W-2)^2$$

And the same for E.

$$E = \pi(W-2)\left(\frac{1}{\tan(30)} - 1\right)$$

$$E_G = \frac{6\mathrm{S}+6\mathrm{E}+12\mathrm{C}+\mathrm{H}}{\frac{3}{2}\sqrt{3}W^2\pi} = \frac{6(W-2)\left(\frac{4}{3}\pi\right)+6\pi(W-2)\left(\frac{1}{\tan(30)}-1\right)+13.4\pi+\pi\frac{3\sqrt{3}}{2}(W-2)^2}{\frac{3}{2}\sqrt{3}W^2\pi} \qquad 7\text{-}8$$

Appendix B: Code

This section contains all the code that has been used to estimate values for the implicit solution and location privacy. All methods are Monte Carlo-based.

Monte Carlo method implicit solution aligned rectangles

Code used to calculate the expected value for the number of guesses for non-aligned rectangles

```
#include <time.h>
#include <cmath>
#include <cstdio>

int loop = 5000000;
int count = 0;
double R = 0.5;
double pi = 3.14149265;

int main() {
  srand((unsigned int)time(NULL));

  double x1;
  double y1;
  double x2;
  double y2;

  for(int i = 0; i < loop; i++) {
    x1 = (double) rand() / ((double) RAND_MAX*(double)2);
    y1 = (double) rand() / ((double) RAND_MAX*(double)2);

  do {
      x2 = ((double) rand() / ((double) RAND_MAX/(double)2)) - 1;
      y2 = ((double) rand() / ((double) RAND_MAX/(double)2)) - 1;
    }while(sqrt(pow(x1-x2,2)+pow(y1-y2,2)) > R); //Generate new values if the point is not in prose
range

    double A = (pi*(2*pi-pi/2-acos(1-x1)-acos(1-y1))/(2*pi)+sqrt(1-pow(1-x1,2))*(1-x1)/2+sqrt(1-
pow(1-y1,2))*(1-y1)/2+(1-x1)*(1-y1));
    double B = pi - (acos(1-y1)-sqrt(1-pow(1-y1,2))*(1-y1)) - ((pi*(2*pi-pi/2-acos(1-x1)-acos(1-
y1))/(2*pi)+sqrt(1-pow(1-x1,2))*(1-x1)/2+sqrt(1-pow(1-y1,2))*(1-y1)/2+(1-x1)*(1-y1)));
    double C = pi - (acos(1-x1)-sqrt(1-pow(1-x1,2))*(1-x1)) - ((pi*(2*pi-pi/2-acos(1-x1)-acos(1-
y1))/(2*pi)+sqrt(1-pow(1-x1,2))*(1-x1)/2+sqrt(1-pow(1-y1,2))*(1-y1)/2+(1-x1)*(1-y1)));
    double D = (acos(1-x1)-sqrt(1-pow(1-x1,2))*(1-x1)) + (acos(1-y1)-sqrt(1-pow(1-y1,2))*(1-y1)) +
((pi*(2*pi-pi/2-acos(1-x1)-acos(1-y1))/(2*pi)+sqrt(1-pow(1-x1,2))*(1-x1)/2+sqrt(1-pow(1-y1,2))*(1-
y1)/2+(1-x1)*(1-y1))) - pi;

    if(y2 > 0.5 && x2 > 0.5) {
      count += 1;
      if(D < A) {
             count += 1;
      }
      if(D < B) {
          count += 1;
      }
      if(D < C) {
          count += 1;
      }
    }
    else if(x2 > 0.5 && y2 < 0.5) {
      count += 1;

      if(B < A) {
            count += 1;
      }
      if(B < D) {
          count += 1;
      }
      if(B < C) {
          count += 1;
      }
    }
    else if((x2 < 0.5) && (y2 < 0.5)) {
      count += 1;

      if(A < D) {
            count += 1;
      }
      if(A < B) {
          count += 1;
      }
      if(A < C) {
          count += 1;
      }
    }
    else if((x2 < 0.5) && (y2 > 0.5)) {
      count += 1;

      if(C < D) {
            count += 1;
      }
      if(C < B) {
          count += 1;
      }
```

```
      if(C < A) {
        count += 1;
      }
    }
    else {
      printf("The algorithm is incorrect(1)!\n");
      printf("x1: %f, y1: %f.\n", x1, y1);
      printf("x2: %f, y2: %f.\n", x2, y2);
    }
  }

  printf("Expected value: %f\n", (double)count/(double)loop);
  return 0;
}
```

Monte Carlo method implicit solution non-aligned rectangles

Code used to calculate AA:

```
#include <time.h>
#include <cmath>
#include <cstdio>

int loop = 5000000;
int count = 0;
double R = 0.5;
double pi = 3.14159265;

int main() {

srand((unsigned int)time(NULL));

  double x1;
  double y1;
  double x2;
  double y2;

for(int i = 0; i < loop; i++) {

    x1 = (double) rand() / ((double) RAND_MAX*(double)2);
    y1 = (double) rand() / ((double) RAND_MAX*(double)2);

do {
      x2 = ((double) rand() / ((double) RAND_MAX/(double)2)) - 1;
      y2 = ((double) rand() / ((double) RAND_MAX/(double)2)) - 1;
    }while(sqrt(pow(x1-x2,2)+pow(y1-y2,2)) > R); //Generate new values if the point is not in prose range

double N = pi - acos(1-y1)-sqrt(1-pow(1-y1,2))*(1-y1) - (pi*(2*pi-pi/2-acos(1-x1)-acos(1-y1))/(2*pi)+sqrt(1-pow(1-x1,2))*(1-x1)/2+sqrt(1-pow(1-y1,2))*(1-y1)/2+(1-x1)*(1-y1));
    double M = (pi*(2*pi-pi/2-acos(1-x1)-acos(1-y1))/(2*pi)+sqrt(1-pow(1-x1,2))*(1-x1)/2+sqrt(1-pow(1-y1,2))*(1-y1)/2+(1-x1)*(1-y1));
    double P = acos(1-x1)-sqrt(1-pow(1-x1,2))*(1-x1);

if(y2 > 0.5) {
      count += 1;
      if(P < M) {
          count += 1;
      }
      if(P < N) {
        count += 1;
      }
    }

else if(x2 > 0.5) {
      count += 1;

      if(N < M) {
         count += 1;
      }
      if(N < P) {
        count += 1;
      }
    }

else if((x2 < 0.5) && (y2 < 0.5)) {
      count += 1;

      if(M < P) {
         count += 1;
      }
      if(M < N) {
        count += 1;
      }
    }

else {
      printf("The algorithm is incorrect(1)!\n");
      printf("x1: %f, y1: %f.\n", x1, y1);
      printf("x2: %f, y2: %f.\n", x2, y2);
    }
  }

printf("Expected value: %f\n", (double)count/(double)loop);
  return 0;
}
```

Monte Carlo method implicit solution hexagons

Code used to calculate the expected value for number of guesses for heaxgon

```
#include <time.h>
#include <cmath>
#include <cstdio>

double hexagon_side = 1.2; //multiples of prose range
int loop = 10000000;
int count = 0;

double prose_range = 0.5;
double v2 = hexagon_side;
double v = v2/2;
double h = v2*0.866;
double R = prose_range;

bool isInside(double, double, double, double);

int main() {

    printf("v2: %f, v: %f, h: %f.\n", v2, v, h);

    srand((unsigned int)time(NULL));

    double x1;
    double y1;
    double x2;
    double y2;

    for(int i = 0; i < loop; i++) {

        do {
            x1 = ((double) rand() / (double) RAND_MAX) * h;
            y1 = ((double) rand() / (double) RAND_MAX) * v2;
        }while(!isInside(x1, y1, 0, 0));

        do {
            x2 = (((double) rand() / ((double) RAND_MAX/(double)2)) - 1)*(h+R);
            y2 = (((double) rand() / ((double) RAND_MAX/(double)2)) - 1)*(v2+R);
        }while(sqrt(pow(x1-x2,2)+pow(y1-y2,2)) > R);

        if(isInside(x2, y2, 0, 0)) {
            count += 1;
        }

        else if((y1/x1) < 1/sqrt(3)) {
            if(isInside(x2, y2, 2*h, 0)){
                count += 2;
            }
            else if(isInside(x2, y2, h, 3*v)){
                count += 3;
            }
            else {
                printf("The algorithm is incorrect(1)!\n");
                printf("x1: %f, y1: %f.\n", x1, y1);
                printf("x2: %f, y2: %f.\n", x2, y2);
            }
        }

        else if((y1/x1) < sqrt(3)) {
            if(isInside(x2, y2, h, 3*v)){
                count += 2;
            }
            else if(isInside(x2, y2, 2*h, 0)){
                count += 3;
            }
            else {
                printf("The algorithm is incorrect(2)!\n");
                printf("x1: %f, y1: %f.\n", x1, y1);
                printf("x2: %f, y2: %f.\n", x2, y2);
            }
         }
        else {
            if(isInside(x2, y2, h, 3*v)){
               count += 2;
            }
            else if(isInside(x2, y2, -h, 3*v)){
               count += 3;
            }
          else {
               printf("The algorithm is incorrect(3)!\n");
               printf("x1: %f, y1: %f.\n", x1, y1);
               printf("x2: %f, y2: %f.\n", x2, y2);
            }
         }
      }
   printf("Expected value: %f\n", (double)count/(double)loop);
   return 0;
}
bool isInside(double x, double y, double hex_center_x, double hex_center_y) {
   x = std::abs(x - hex_center_x);
   y = std::abs(y - hex_center_y);

   if (x > h || y > v2) return false;

   return (2 * v * h - v * x - h * y) >= 0;
}
```

Monte Carlo method location privacy truncation

```
#include <time.h>
#include <cmath>
#include <cstdio>
int loop = 1000000;
double R = 1;
double pi = 3.14159265;
void locpriv(double, double);
int main()
{
            srand((unsigned int)time(NULL));
            double W = 3.6;
            double H = W;
            for (double j = 2; j < 10.1; j += 0.4) {
                        locpriv(j / sqrt(2), j / sqrt(2));

            }

            return 0;

}
void locpriv(double W, double H) {
            double x1, y1, x2, y2, x3, y3, x4, y4, x5, y5;
            double average = 0;

            for (int i = 0; i < loop; i++) {

                        //UE1 location
                        x1 = ((double)rand() / (double)RAND_MAX) * 2 * W;
                        y1 = ((double)rand() / (double)RAND_MAX) * 2 * H;

                        //UE2 location
                        do {
                                x2 = x1 + (((double)rand() / ((double)RAND_MAX / (double)2)) - 1);
                                y2 = y1 + (((double)rand() / ((double)RAND_MAX / (double)2)) - 1);
                        } while (sqrt(pow(x2 - x1, 2) + pow(y2 - y1, 2)) >= R);

                        //Obfuscated UE1 location
                        x3 = ((int)(x1 / W))*W;
                        y3 = ((int)(y1 / H))*H;

                        //Estimated UE1 location
                        x4 = x3 + W / 2;
                        y4 = y3 + H / 2;

                        //With proximity constraint
                        do {

                                x5 = x4 + (((double)rand() / ((double)RAND_MAX / (double)2)) - 1)*W / 2;
                                y5 = y4 + (((double)rand() / ((double)RAND_MAX / (double)2)) - 1)*H / 2;
                        } while (sqrt(pow(x5 - x1, 2) + pow(y5 - y1, 2)) > R);

                        average += sqrt(pow(x5 - x2, 2) + pow(y5 - y2, 2));

            }

            average /= loop;
            printf("Average(W=%f, H=%f): %f.\n", W, H, average);

}
```

Monte Carlo method location privacy random rectangle

```
#include <time.h>
#include <cmath>
#include <cstdio>

int loop = 10;
double R = 1;
double pi = 3.14159265;

void locpriv(double, double);

int main()
{

srand((unsigned int)time(NULL));


  for(double j = 1; j < 10.1; j+=0.4) {

        locpriv(j, j);

  }
  return 0;
}
void locpriv(double W, double H) {
   double x1, y1, x2, y2, x3, y3, x4, y4;
   double average=0;
   for(int i = 0; i < loop; i++) {
         do {
              //UE2 location
         x1 = ((double) rand() / ((double) RAND_MAX/(double)2)) - 1;
         y1 = ((double) rand() / ((double) RAND_MAX/(double)2)) - 1;
      }while(sqrt(pow(x1,2)+pow(y1,2)) > R);
      x2 = (((double) rand() / ((double) RAND_MAX/(double)2)) - 1)*W;
      y2 = (((double) rand() / ((double) RAND_MAX/(double)2)) - 1)*H;
      x3 = x1 + x2;
      y3 = y1 + y2;
      do {
      x4 = x3 + (((double) rand() / ((double) RAND_MAX/(double)2)) - 1)*W;
      y4 = x3 + (((double) rand() / ((double) RAND_MAX/(double)2)) - 1)*H;
      } while(sqrt(pow(x4, 2)+pow(y4, 2)) > R);
      average += sqrt(pow(x4-x1,2)+pow(x4-y1,2));
   }
   average /= loop;
   printf("Average(W=%f, H=%f): %f.\n", W, H, average);
}
```

Monte Carlo method location privacy random circle

```
#include <time.h>
#include <cmath>
#include <cstdio>
int loop = 10;
double R = 1;
double pi = 3.14159265;
void locpriv(double, double);
int main()
{


   srand((unsigned int)time(NULL));
   for(double j = 1; j < 10.1; j+=0.4) {
      locpriv(j, j);
   }
```

```
  return 0;

}

void locpriv(double W, double H) {

   double x1, y1, x2, y2, x3, y3, x4, y4;
   double average=0;

   for(int i = 0; i < loop; i++) {

      do {
         //UE2 location
         x1 = ((double) rand() / ((double) RAND_MAX/(double)2)) - 1;
         y1 = ((double) rand() / ((double) RAND_MAX/(double)2)) - 1;
     }while(sqrt(pow(x1,2)+pow(y1,2)) > R);

      do {
         x2 = (((double) rand() / ((double) RAND_MAX/(double)2)) - 1)*R2;
         y2 = (((double) rand() / ((double) RAND_MAX/(double)2)) - 1)*R2;
      } while(sqrt(pow(x2, 2)+pow(y2, 2)) > R2);

      x3 = x1 + x2;
      y3 = y1 + y2;

      do {
         x4 = x3 + (((double) rand() / ((double) RAND_MAX/(double)2)) - 1)*W;
         y4 = x3 + (((double) rand() / ((double) RAND_MAX/(double)2)) - 1)*H;
     } while(sqrt(pow(x4, 2)+pow(y4, 2)) > R);

     average += sqrt(pow(x4-x1,2)+pow(x4-y1,2));

   }

average /= loop;
printf("Average(W=%f, H=%f): %f.\n", W, H, average);

}
```

TRITA-ICT-EX-2016:67