# PRIVACY PROTECTION FOR SPATIAL TRAJECTORIES AGAINST BRUTE-FORCE ATTACKS

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

MASTER OF SCIENCE

IN

COMPUTER ENGINEERING

By

Dorukhan Arslan

August 2018

PRIVACY PROTECTION FOR SPATIAL TRAJECTORIES AGAINST BRUTE-FORCE ATTACKS

By Dorukhan Arslan

August 2018

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Erman Ayday(Advisor)

Fazlı Can

Ali Aydın Selçuk

Approved for the Graduate School of Engineering and Science:

Ezhan Karaşan
Director of the Graduate School

# ABSTRACT

## PRIVACY PROTECTION FOR SPATIAL TRAJECTORIES AGAINST BRUTE-FORCE ATTACKS

Dorukhan Arslan
M.S. in Computer Engineering
Advisor: Erman Ayday
August 2018

The prevalence of Global Positioning System (GPS) equipped mobile devices and wireless communication technologies have resulted in widespread development of location-based services (LBS). As some typical examples of LBS, routing, tracking, local search, social networking, and context advertising can be given. In terms of update frequency of location, LBS are divided into two categories: snapshot and continuous. Snapshot LBS request a user's location only once to control features. Continuous LBS, on the other hand, require a user's location in a dynamically periodic or on-demand manner. In the course of interaction with a continuous LBS application, the user reveals a sequence of location samples, namely, spatial trajectory, to service provider. Trajectory privacy in such services is of great importance, since adversaries may use the spatio-temporal sequential pattern to disclose the user's personally identifiable information (PII) with high certainty. In order to prevent this from happening, service providers generally encrypt spatial trajectory data under the user's password, and then store in their databases. However, potential adversaries may decrypt the encrypted database via a brute-force attack. In other words, they try every possible value for a password until success is achieved. Although using high-entropy passwords have caused inconvenience for adversaries, the encryption schemes of service providers are vulnerable to this type of an attack due to the tendency of users to choose weak passwords. Also, if the rapid evaluation of computing technology and algorithmic advances are taken into consideration, even the use of a large password domain with conventional encryption can lead to the success of a brute-force attack that became feasible computationally. Thus it is crucial to assess privacy threats and take security countermeasures for spatial trajectories.

We present a system that incorporates honey encryption (HE) scheme that

provides security beyond the brute-force bound in order to offer absolute protection for spatial trajectories against data breaches that involve computationally unbounded adversary. Our technique guarantees that decryption under any password will yield a plausible-looking trajectory. If an adversary decrypts an encrypted trajectory with a wrong password, it cannot eliminate that password, since the system returns an incorrect trajectory that is impossible to distinguish from the correct one. To efficiently encode and decode a spatial trajectory, we build a precise tree-based distribution transforming encoder (DTE) as the fundamental requirement of HE. In addition, we introduce the methods to dynamically update the proposed DTE. To prove the security guarantee of our system, we evalute it considering several attacks with and without side information using a real-life GPS sampling data set taken from 537 taxis over 30 days.

*Keywords:* Spatial Trajectory, Location-Based Service, Location Privacy, Honey Encryption.

# ÖZET

# UZAMSAL GEZİNGELERİN KABA GÜÇ SALDIRILARINA KARŞI GİZLİLİK KORUMASI

Dorukhan Arslan
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Danışmanı: Erman Ayday
Ağustos 2018

Küresel Konumlandırma Sistemi (GPS) donanımlı mobil cihazlar ve kablosuz iletişim teknolojilerinin yaygınlaşması konum tabanlı hizmetlerde (LBS) geniş alana yayılmış bir gelişime sebep oldu. Bölgesel arama, rotalama, konum takibi, sosyal paylaşım ve bağlamsal reklam LBS'lere örnektir. Konum verisi toplama sıklıklarına göre LBS'ler anlık ve sürekli olmak üzere ikiye ayrılır. Anlık LBS söz konusu hizmeti sunabilmek için kullanıcının konum verisini bir kez iletmesine ihtiyaç duyar. Öte yandan, sürekli LBS kullanıcının konumunu periyodik olarak ya da her talep edildiğinde servis sağlayıcısı ile paylaşmasını gerektirir. Bir sürekli LBS sunan uygulamanın kullanımı sırasında, kullanıcı servis sağlayıcısına konum verilerinin birbiri ardına sıralanmasından oluşan kayıt listesini, yani uzamsal gezingesini, iletir. Sisteme saldırı düzenleyecek kötü niyetli kimseler, kullanıcılara ait uzamsal-zamansal dizi modellerinden faydalanarak şahısları tanımlamak için kullanılan bilgilere (PII) yüksek kesinlikte ulaşabilmesinden ötürü, bu servislerde tutulan gezingelerin gizliliği son derece önem taşımaktadır. Bu gibi durumların önüne geçmek amacıyla, servis sağlayıcıları genellikle uzamsal gezingeleri kullanıcı parolasıyla şifreledikten sonra veri tabanlarında kayıt altında tutmaktadırlar. Ancak, potansiyel bir saldırgan şifrelenmiş veri tabanını bir kaba kuvvet saldırı vasıtasıyla deşifre edebilir. Başka bir deyişle, kullanıcılara ait gezingelerine ulaşana kadar olabilecek tüm parola kombinasyonlarını deneyebilirler. Her ne kadar yüksek entropili parola kullanımı saldırganların işini güçleştiriyor olsa da kullanıcılar zayıf parola seçme alışkanlıkları nedeniyle servis sağlayıcıların şifreleme şemaları bu tip saldırılara karşı zaafiyet taşımaktadır. Ayrıca, hesaplama teknolojilerinin ve ilgili algoritmaların hızlı gelişimi göz önünde bulundurulduğunda ne kadar geniş bir parola aralığı seçilse de kaba kuvvet saldırıları istatistiki olarak başarıyla sonuçlanabilmektedir. Bu sebeplerden ötürü, uzamsal gezingelerin gizliliği tehdit eden unsurların incelenip gerekli güvenlik

önlemlerinin alınması son derece gereklidir.

Bu doğrultuda uzamsal gezingelere saldırı düzenleyen hesaplama sınırı bulunmayan şahısların neden olacağı veri ihlallerine karşı mutlak koruma sağlamak amacıyla, kaba kuvvet saldırıları limitinin ötesinde bir koruma sağlayan honey encryption (HE) ile beraber çalışan bir sistem sunuyoruz. Tekniğimiz şifrelenmiş bir uzamsal gezingenin deşifre edilmesi sonucunda her durumda makul bir görünüme sahip gezingeye ulaşılmasını garanti etmektedir. Bu demektir ki bir saldırgan şifrelenmiş bir gezingeyi yanlış bir şifre deneyerek deşifre ettiğinde, bu şifrenin yanlışlığını doğrulayamayacak, çünkü sistem bu saldırgana gerçeğinden ayırt etmenin mümkün olmadığı sahte bir gezingeyi sonuç olarak verecektir. Bir uzamsal gezingeyi etkin bir şekilde kodlama ve gezingenin kodlanmış halini geri çözmek için, ağaç tabanlı bir dağıtım dönüştürücü kodlayıcı (DTE) oluşturarak HE uygulamak için en temel gereksinimi yerine getirdik. Buna ek olarak, DTE ağacını dinamik olarak yenilememize olanak sağlayacak metotları tanıttık. Sistemimizin güvenlik garantisini ispat etmek için, potansiyel bir saldırganın ulaşmaya çalıştığı verilerle ilgili yan bilgisinin olduğu ve olmadığı çeşitli saldırı senaryolarını sistemi 537 taksiden 30 gün boyunca toplanmış gerçek bir GPS veri seti üzerinde uygulayarak analiz ettik.

*Anahtar sözcükler*: Uzamsal Gezinge, Konum Tabanlı Servis, Konum Gizliliği, Honey Encryption.

# Acknowledgement

I would like to acknowledge and thank the following important people who have supported me, not only during the course of the thesis, but throughout my Master's degree.

First of all, I would like to express my thanks and sincere appreciation to my supervisor Dr. Erman Ayday for the encouragement, creative and comprehensive advice throughout the research and degree.

I would like to thank Aykut Güven from Bilkent University for his companion and proofreading. Thanks to his encouragement and belief in me, I could stay on focus on this hugely rewarding and enriching progress.

I would also like to thank Dr. Ayday's research group, especially Didem Demirağ, and classmates for their kindness and support.

And lastly, I would like to thank my mother and father, for their moral and guidance in order to finish this study.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

The emerging convergence and integration of digital communication technology based on mobile networks raised the importance of information on the geographical location of mobile devices. Today location is one of the most important aspects of context in pervasive computing. People increasingly tend to use mobile devices to share their whereabouts with third-parties in return for location-based services (LBS). LBS constantly innovate and endeavor to have complete user satisfaction. It changed the way people transact business and organize their activities and free time by creating a dynamic user experience that adds value and convenience [1]. LBS market has grown considerably over the past few years and is expected to grow further. According to a new report, the market is expected to reach $61,897 million by 2022 [2]. In despite, however, of the advantages of LBS, these services necessarily require the collection of significant amounts of users' spatio-temporal data to provide location-specific information, and the data can be used to harm individuals if it falls in wrong hands.

The update frequency of a user's location varies depending on the offered service. As the number of location data that a user should reveal to service provider increases, an adversary's chance to harm that user increases as well. A recent study showed that four spatio-temporal points are enough to uniquely identify 95% of 1.5 million people in a mobility database even when the resolution of the

data set is low [3]. There are two main LBS types in terms of update frequency of location: snapshot and continuous. A snapshot LBS requests a user's location only once such as sporadic queries that are performed relying on self-reported positioning. An individual obtains information from service provider when its current position is transferred. As a typical example, a point-of-interest query to find nearest restaurant can be given. On the other hand, a continuous LSB requires a user's location in a dynamically periodic or on-demand manner. In the course of interaction with a continuous LBS, the user reveals a sequence of location samples to service provider. To exemplify continuous LBS, position awareness services used for monitoring an individual's position (e.g., in-car navigation systems and GPS-enabled PDAs) or location tracking services that receive periodical and frequent updates of an individual's location (e.g., mobile highway telematics systems for estimating traffic congestion) [4] can be given.

In continuos LBS, the receiver of the location updates accumulates prior movement data for several purposes such as assessing traffic [5], training a system about a user's habits [6], creating customized driving routes [7], helping predict where a user is going [8], or creating a travelogue [9]. To put it in a different way, the service provider collects sequences of highly frequent location reports that has come from consumers of its service in a spatial database, and exploits this data if the user gave permission. Commonly, these sequences, namely spatial trajectories, are stored by service providers after encryption with password-based encryption (PBE) under user passwords. However, multipurpose use of a large volume of location data from several individuals still opens a door to critical privacy issues even for such systems that employ PBE [10]. Users prevalently tend to choose weak passwords [11], and this common tendancy makes such systems vulnerable to brute-force attacks. An adversary can use spatial trajectories to disclose the user's personally identifiable information (PII) with high certainty, if it has defeated the encryption or access control on the data. It may harm a person economically, invite unwelcome advertisements, enable stalking or physical attacks or infer embarrassing proclivities [12, 13]. For example, if the starting location point of a trajectory is home, and adversary can use reverse geocoding to get the home address that is associated with that location point. Then, it can use a

people-search-by-address engine to find the residents of the home address [14].

To assess privacy threats and countermeasures for spatial trajectories, the common encryption-based approach is encrypting spatial trajectories using a conventional PBE. In theory, it is computationally impossible to defeat a PBE scheme with sufficiently large password space. In practice, however, since a great majority of users in a system choose low-entropy passwords [15], PBE scheme can be defeated via a brute-force attack. As a major upgrade to PBE, Juels and Ristenpart introduced a theoretical framework for encryption called honey encryption (HE) that gives a plausible-looking yet incorrect plaintext when a ciphertext is decrypted with an incorrect cryptographic key or password [16]. In other words, HE does not only provide data cryptographic protection, it also gives an additional layer of protection by serving up fake data in response to the adversary's every incorrect password guess [10]. Hence, the adversary cannot realize if it tries correct or incorrect password, and cannot eliminate any password from the set of possible passwords by trial and error. Thanks to this property, HE scheme provides a security beyond the brute-force bound and thus it outperforms PBE. Nevertheless, since HE relies on a highly accurate distribution-transforming encoder (DTE) to transform message space to seed space, applying HE in LBS databases that store spatial trajectories is a non-trivial task because it requires a quantitative understanding of the message space [10]. However, existing methods do not offer any practical solution to efficiently apply HE to complex message domains such as spatial trajectories. To put it simply, building a DTE which presents the precise probability of every possible message is the fundamental problem of modeling HE for complicated real-life data including spatial trajectories. The characteristics of spatial trajectory data is not uniformly distributed, and for this reason, it is hard to comprehend and define a DTE scheme for spatial trajectories systematically. In addition, HE is applied previously to the data collections that are not updated over time but LBS providers maintain databases that need to be dynamically updated on a regular basis. These issues are the main challenges we tackle throughout this thesis.

Consequently, we propose a framework which utilizes HE to address the problem of protecting spatial trajectory data in continuous LBS. Our framework secures a system regardless of the size of chosen password space and the user's tendancy to choose a low entropy password. The method we implemented acts honestly to a user that has a authenticated password and responds the correct spatial trajectory. It acts deceivingly to an adversary that attempts a brute-force attack and responds a plausible-looking but fake spatial trajectory.

The contributions of this work can be summarized as follows:

- We introduce a new model to address the problem of protecting spatial trajectories against security breaches in which an adversary with unbounded computation capability is involved, and develop a technique to store and retrieve spatial trajectories in a secure way.
- We provide an extension for HE to apply to the databases that are updated over time on a regular basis.
- We provide a formal security analysis of our proposed techniques.
- We implement our proposed techniques and show their efficiency.
- We tested our model under different settings, and show its security guarantee against an adversary with side information in order to decrypt spatial trajectories.

The rest of thesis is organized into 6 sections. In Chapter 2, a background information related to spatial trajectory data, brute-force attacks, HE, and PBE is provided. Moreover, related work is reviewed. In Chapter 3, the formal definition of the problem is given and the proposed techniques are described in detail. In Chapter 4, the performance of the proposed model is evaluated with different scenarios. In Chapter 5, the security of the system is analyzed considering an adversary with side information as well. Finally, in Chapter 6, the thesis is concluded.

# Chapter 2

# Background and Related Work

In this chapter, the core concepts that our proposed method is based on are outlined as the background knowledge. The chapter gives information about brute-force message-recovery, honey encryption (HE), and password-based encryption (PBE) respectively. The chapter is concluded with related work.

## 2.1 Brute-Force Attacks

Brute-force attack is a trial-and-error method used by automated software to decode encrypted data such as passwords through exhaustive effort rather than employing intellectual strategies. It is also known as exhaustive key search or brute-force cracking. In a brute-force attack, the automated software is used to generate a large number of possible combinations of legal characters in sequence as a set of guesses of the desired data. Then, it is configured to proceed through these guesses until a combination becomes statistically correct and cracks the code. Assuming the output of encryption of a message $M$ from message distribution $p_m$ under a key $K$ from password distribution $p_k$ is ciphertext $C$, an adversary that use brute-force attack intends to get $M$ decrypting $C$ by trying necessary number of possible keys.

Conventional password-based encryption (PBE) methods strengthen a brute-force attacker's hand, since it gives a high chance of eliminating incorrect passwords [17]. Although a brute-force attack is supposed time and resource-consuming by nature, it is an infallible approach. The necessary amount of time to break a cipher is proportional to the size of the secret key. The maximum number of attempts is equal to $2^{keysize}$, where $keysize$ is the number of bits in the keys. It means that the time to break a cipher and obtain the desired data may be increased if more advanced encryption schemes are applied; notwithstanding, the success of the adversary is usually based on the number of combinations tried. The number of possible passwords that have to be tested can be significantly reduced if some practical strategies (e.g., dictionary attacks and reverse brute-force attacks) are implemented by the adversary that performs an automated attack. Moreover, the fast-paced advances in computational power of computers and parallel computing techniques makes even the most exhausting cryptographic brute-force attacks scalable processes with each passing day.

Brute-force attacks are a serious threat capable of affecting great number of users and services. In 2013, GitHub notified several users about potentially being a victim of a brute-force attack. Although GitHub aggresively rate-limit login attempts and passwords are stored properly, the incident was inevitable. A considerable number of users had weak passwords that led to the site being targeted and hence sensitive data of these users are obtained by the attackers. Due to the countermeasures taken by GitHub security, the attackers could achieve to fly under the radar, since they used over 40,000 unique IP addresses and the attack was done slowly on purpose in order to not raise any alarm. As result, GitHub gave information about the new rate-limiting measures, and notified users that they would no longer be able to login to the site with commonly-used weak passwords [18]. Considering this example, it is obvious that PBE method does not provide enough security for the encrypted data against brute-force attacks even if the service provider takes some countermeasures. In order to eliminate the threat of brute-force attacks on the systems that are protected with PBE, replacing PBE with an encryption method that provide security beyond the brute-force bound is essential.

## 2.2 Honey Encryption (HE)

Honey encryption (HE) [16], introduced by Juels and Ristenpart, is an encryption paradigm designed to produce ciphertexts which yield fake but plausible-looking, namely honey, plaintext upon decryption with wrong keys. Thus it provides security against an attacker or another malicious intended user who is carrying out a brute-force attack to know she has correctly guessed a password or encryption key. Usually the term "honey" is used to indicate a decoy that is used to attract the adversary. HE has a lot in common with honeypots in the sense that both are used to distract and defend against attackers. Honeypots are used to detect unauthorized attempts to the information system. HE, on the other hand, is basically an encryption technique which provides a deflection mechanism in which even a computationally unbounded adversary cannot realize if a message is correct or honey. It is used to detect attackers when trying to decrypt the encrypted data.

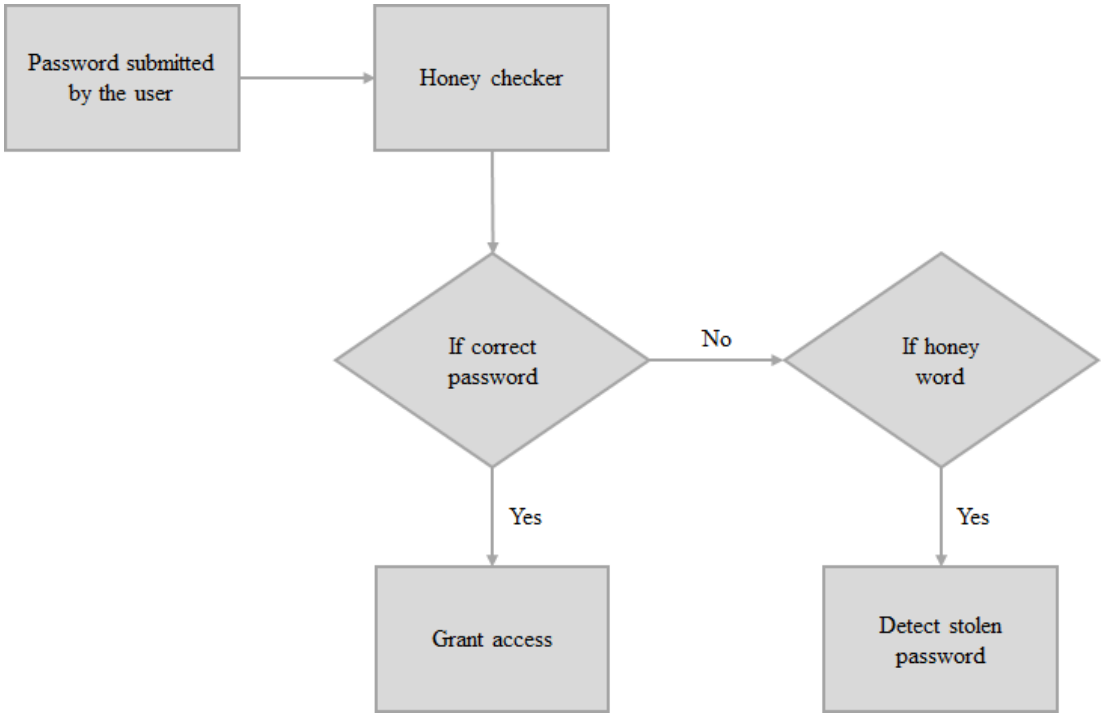


Figure 2.1: The authentication process in HE.

In a conventional cryptographic system, when an adversary decrypts a ciphertext using a wrong key, an invalid message is responded. The adversary can infer that a key is wrong since the decrypted results will be unintelligible. These systems give advantage to an adversary to eliminate wrong keys via a brute-force attack. But in case of HE, there is a list of passwords, and all are honey except only one which is right. If an adversary uses brute-force to the encryption, as long as it is providing input, a password that mimics a real looking one but actually fake is outputted. Hence, the adversary confuses and cannot distinguish the real password from the large number of generated fake passwords. Moreover, when an adversary tries to decrypt by guessing the password, the system that encrypts the database under a HE scheme can flag that adversary for trying honey words. The mentioned authentication process is visualized in Figure 2.1.

| | |
|---|---|
| $\mathcal{M}$ | Message space |
| $p_m$ | Message distribution |
| $M$ | A message of spatial trajectory, $M \in \mathcal{M}$ |
| $\mathcal{K}$ | Password (key) space |
| $p_k$ | Password (key) distribution |
| $K$ | A user password, $K \in \mathcal{K}$ |
| $\mathcal{S}$ | Seed space |
| $\mathcal{C}$ | Ciphertext space |
| $p_d$ | DTE message distribution |

Table 2.1: Notations and definitions of the proposed scheme.

HE is used to tailor encryption schemes to specific message distributions. It provides a security beyond the brute-force bound. To present the concept more formally, suppose $M$ is a message that is sampled from a message distribution $p_m$ over the message space $\mathcal{M}$, and $C \in \mathcal{C}$ is a ciphertext that is encrypted under key $K \in \mathcal{K}$ by using HE. When $C$ is decrypted under an incorrect key $K' \neq K$, the message $M'$ from the distribution $p_m$ will be plausible-looking but fake. Therefore, the adversary cannot use the advantage of eliminating wrong keys.

The main component of HE framework, distribution-transforming encoder (DTE), is described below.

## Distribution-Transforming Encoder (DTE)

The approach taken to realize HE uses a pair of algorithms represented as $DTE = (encode, decode)$ in order to transform a non-uniform message distribution $p_m$ into a uniform distribution over a seed space $\mathcal{S}$. *encode* is a probabilistic algorithm that takes a message $M \in \mathcal{M}$ as input and outputs a value in a set $\mathcal{S}$, the seed space. *decode* takes as input a value $S \in \mathcal{S}$ and outputs a message $M \in \mathcal{M}$. Basically, *encode* maps a message to a point in a set, whereas *decode* reverses the encoding. Since the encoding algorithm is probabilistic, mapping of a message may not correspond to a unique seed. A given message $M$ can potentially be mapped to a set of seeds $S_M \subseteq \mathcal{S}$, where $S_M$ has to include one seed at least. For different messages $M$ and $M'$, $S_M \cap S'_M = \emptyset$, and $\bigcap_{M \in \mathcal{S}} S_M = S$. Nevertheless, *decode* is deterministic. To put it simply, a given seed $S \in S_M$ is enough to generate the message $M$. In the light of this information, we lead to the conclusion that it is an important attribution for DTE, $Pr[decode(encode(M)) = M] = 1$.

In DTE-then-encrypt construction in HE [16], messages are encoded through the DTE at first, then encrypted under a conventional symmetric encryption (SE) scheme using a key derived from the password space $p_m$. Decryption consists of two steps as well: the decryption algorithm of the SE scheme is applied at first, and then the DTE decoding algorithm outputs the message. Given a secure DTE, an adversary cannot distinguish a pair $(M, S)$ generated by selecting $M$ from $p_m$ and encoding it to obtain seed $S$, and a pair $(M, S)$ generated by selecting a seed $S$ uniformly at random and decoding it to obtain message $M$.

The HE construction is shown in 2.2. Let $M \subseteq \mathcal{M}$, $K \subseteq \mathcal{K}$, $S \subseteq \mathcal{S}$, and $C \subseteq \mathcal{C}$. $\mathcal{S} = \{0,1\}^l$ denotes the seed space with length $l$. The conventional symmetric encryption scheme $SE = (encrypt, decrypt)$ uses random bits uniformly sampled from $\{0,1\}^B$ during encryption. The HE setup $HE[DTE, SE]$ consists of a pair of algorithms, $HEnc$ and $HDec$, for encryption and decryption respectively. For a message $M$ under a key $K$, $HEnc(K, M)$ outputs a ciphertext $C$ and a random salt $r$ of length $B$. $HDec(K, (r, C))$ outputs message $M$ using $K$ and $(r, C)$.

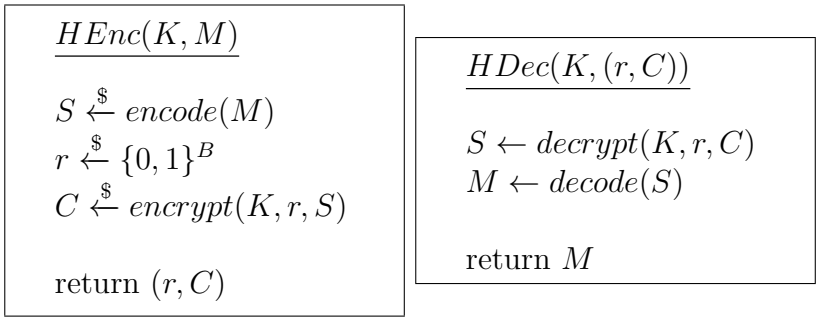


Figure 2.2: DTE-then-encrypt construction using a symmetric encryption. The symbol '$' over an arrow implies randomness of the function.

## 2.3 Password-Based Encryption (PBE)

Password-based encryption (PBE) [17] is a form of symmetric-key generation that typically takes a low-entropy, user-supplied password as input, adds some entropy to it, and then generates a strong secret key using several data-scrambling techniques. The generated key can then be used for symmetric encryption. In other words, that key can be used for both encryption and decryption of the input string. There are two popular PBE standards: PKCS #5 and PKCS #12. PKCS #5 supports ASCII characters as input string. On the other hand, PKCS #12 supports 16-bit characters.

The strength of the cipher directly depends on the strength of the secret key. A strong secret key must not be predicted with ease. The key bytes are supposed to be as random and unpredictable as possible. Since passwords are generally memorable subsets of ASCII or UTF-8 characters, a secret key cannot be derived from the password provided by the user. For this reason, PBE algorithms do not only use a user's password but also some additional input parameters, salt and quantity of iterations.

As a general rule, passwords are not stored in plaintext, but rather hashed. An adversary can simply generate a table of common passwords and their corresponding hashes. If users select common passwords, it is trivial to reveal the passwords exploiting this precalculated table for the adversary. A salt is a random number that is added to make a common password less predictable. The

salt lowers the probability that the hash-value will be found in the table if it is combined with the password. It is possible to store the salt in the clear in the database with the hashed value, hence generating the table for each salt and check all the likely PBE algorithm inputs is a costly operation. Moreover, since it is highly unlikely that the same salt would be created by a pseudorandom number generator, a salt may be transmitted along with the ciphertext to the receiver.

PBE algorithms use a mixing function based around a secure hash function to make the key derivation procedure more complicated and time consuming. The function is applied to input a specified number of times. This iteration process causes a delay which is acceptable for a user. However, in point of the adversary, performing authentication procedure for each combination in the table is an intimidating task due to the increased time complexity of a brute-force attack. Unfortunately, it should be mentioned that even the additional input parameters, salt and quantity of iterations, will not make much difference if the password is not sufficiently complex.

## 2.4 Related Work

Location and spatial trajectory data privacy protection in continuous LBS has drawn a lot of attention from the research community and industry in recent years. In the literature, numerous privacy breaches have been proposed to illustrate how to breach the user privacy to obtain trajectory data [19, 20, 21, 22]. There are also several works which addresses issues about providing security and privacy for spatial trajectory data. Chow et al. [14] and Zheng. et al. [23] raise concern about trajectory privacy in LBS and data publication, and emphasize that protecting user location privacy for continuous LBS is more challenging than snapshot LBS due to the user's location information might be inferred by adversaries with higher certainty using the spatial and temporal correlations in the user's location samples. In this paper, however, privacy concerns related with publishing location trajectories to the public or a third party for data analysis are studied rather than privacy concerns against brute-force attacks to retrieve

and store the spatial trajectory data. Similarly, Terrovitis et al. [24] address the problem of protecting privacy in the publication of location trajectories as well. This work shows that an adversary can use partial trajectory knowledge as a quasi-identifier for the remaining location in the sequence, and proposes a data suppression technique to prevent any privacy breach while keeping the published data as accurate as possible. Hasan at al. [25] propose a privacy architecture with a bounded perturbation technique to preserve user trajectories from privacy breaches in LBS applications.

The majority of works about privacy and security of spatial trajectories come up with anonymization techniques to provide security for trajectories without putting forth a complete cryptographic system. The main focus of these works is generally about publication of the data. Hence, they could not avail against computationally unbounded adversaries that will perform brute-force attack to a spatial trajectory database. In addition to these, there are some studies in the literature which use a different strategy called "honey" that purposes to deceive adversaries. For instance, Honeytokens [26] and Honeypots [27] are useful techniques for detection, deflection, and alarming malicious attempts to log in a system. Honeyword [28] is simply an incorrect password which is published as a part of a honeypot. Any user who attempts to log in using a honeyword sets off an alarm, and the adversarial attack has been reliably detected.

As a new honey solution, Juels et al. [16] proposed honey encryption to deceive attackers with plausible looking but incorrect passwords. This study that provides a security beyond the brute-force barrier is applied for several domains such as credit card numbers [29]. Hueng et al. adapt the technique to the domain of genomic data [10]. Before this study, honey encryption was merely applicable to uniform datasets but it extended honey encryption to non-uniform datasets by providing secure storage for the genomic data. In the literature, there are some other honey encryption applications. Kim et al. [30] apply honey encryption to instant messaging system to provide protection against eavesdropping over communication. Yoon et al. [31] propose the concept of visual honey encryption and apply the scheme to two dimensional images.

# Chapter 3

# Proposed Solution

We propose a solution based on honey encryption (HE) for the secure storage and retrieval of a user's spatial trajectory data in continuous location-based services (LBS). User privacy must be protected in this type of LBS, which typically hold a trajectory database, since knowing the spatial trajectory collected from a user can reveal far more than a set of latitude and longitude coordinates. It can be exploited to infer many sensitive information about individuals, such as home address, health condition, lifestyle habits, and political attitude [14]. This chapter explains the details of our framework. In Section 3.1, the problem we tackle is described, and our assumptions for the proposed solution are given. Then, an overview of the considered architecture is introduced with potential threats to user privacy in trajectory databases of continuous LBS. Lastly, the protocol is discussed step by step in Section 3.2, emphasizing the encoding and decoding.

## 3.1 Problem Definition

A spatial trajectory is the path or trace that a moving object reports while following through a geographical space as a function of time [14]. Based on this definition, for a continuous LBS, the moving object that reports its spatial

trajectory corresponds to the service user who walks in the serviceable area. In this model, a spatial trajectory is a sequence of a user's whereabouts. It is simply the message that must be securely stored by our system. The proposed framework is not only specialized for storage, it also enables retrieval and update of spatial trajectories in a secure way.

### 3.1.1 Spatial Trajectory Data Representation

A spatial trajectory $Tr$ is basically a set of $n$ time-ordered points, $Tr : p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_n$, where each point $p_i$ consists of a pair of latitude and longitude coordinates $(x_i, y_i)$ and a timestamp $t_i$, i.e., $p_i = (x_i, y_i, t_i)$, where $1 \leq i \leq n$. However, in our framework, since the spatial database uses a grid structure to index points, a point is represented with the identifier of the grid cell in which it is contained instead of two-dimensional geographic coordinates. It is also assumed that points of a spatial trajectory are reported at periodic intervals. In other words, the elapsed time $t_e$ between any consecutive reports is constant and predetermined, $t_e = t_{i+1} - t_i$. Thus a spatial trajectory $Tr$ is represented as $Tr : c_1 \rightarrow c_2 \rightarrow \cdots \rightarrow c_n$, where $c_i$ is the identifier of a cell in which the target user is located at time $t_i$. In addition, $Tr_{i,j}$ represents subsequence of $Tr$ starting from $c_i$ to $c_j$.

### 3.1.2 System Model

Most of the LBS applications have client-server architecture which is a centralized architecture where mobile users directly communicate with the LBS provider [32, 33]. Thus the architecture described in this section consists of two parties: client (user) and server (service provider). We consider a scenario in which encrypted seeds corresponds users' spatial trajectories are stored in service provider's database. Each user chooses a password and the spatial trajectory data is encrypted under the user's password. We assume users are free to choose low-entropy passwords.

Client is responsible for sending the user's spatial trajectory and the request of retrieving the spatial trajectory back as the plaintext to server. On the other hand, server is responsible for providing services based on the spatial trajectory that the user sent. We assume client does not send the geographical location of the mobile device one at a time. The geographical locations are collected for a predetermined period and stored in the local storage of the mobile device as a spatial trajectory to be sent server instead. Client performs password-based encryption and decryption on the seed, and holds encrypted seeds of user's spatial trajectories. Server performs encoding and decoding, and holds no information rather than DTE tree. An overview of the proposed architecture can be seen in Figure 3.1.

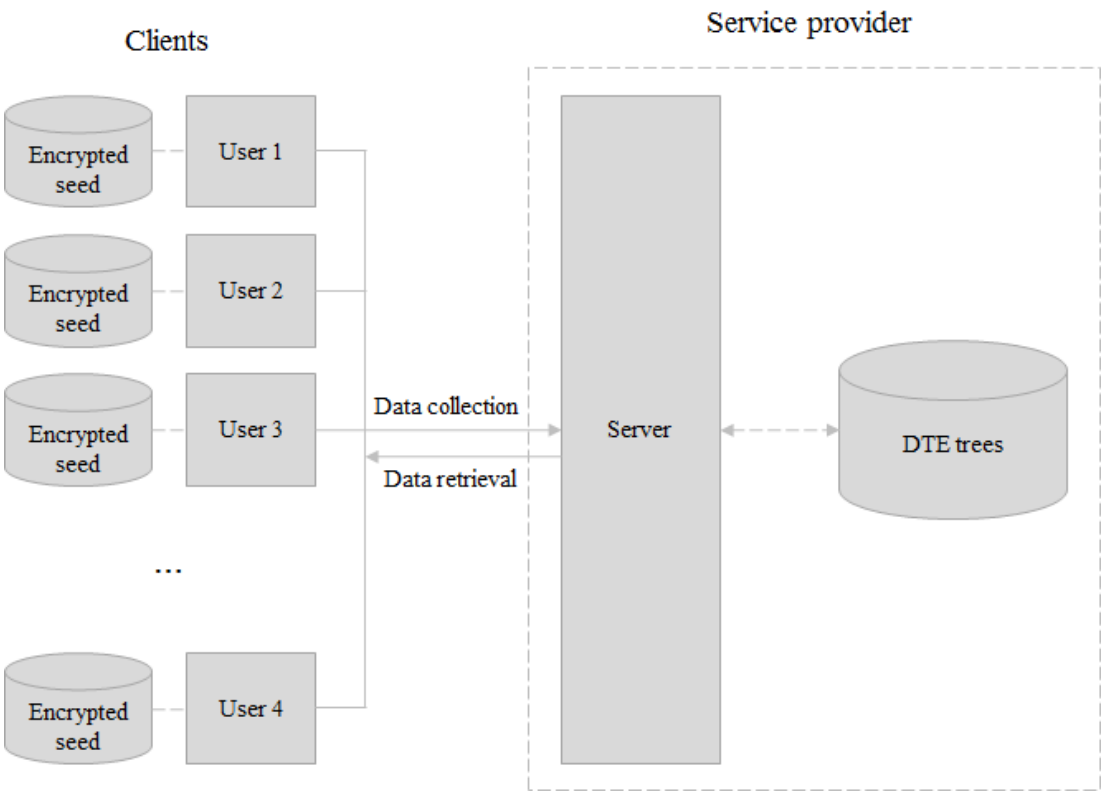


Figure 3.1: System model of spatial trajectory data storage and retrieval.

The system is designed for continuous LBS providers to achieve privacy-preserving storage and retrieval of spatial trajectories. To exemplify a continuous

LBS application that can benefit from such a system, GPS fitness-tracking applications can be given. These applications offer a range of LBS that track users' outdoor and indoor movements with and without wearable devices. On one hand, the service provider exploits spatial trajectories collected from users to improve its services (e.g., finding a jogging route between two locations). On the other hand, a user can display (and share) former routes, namely, spatial trajectories, with some metrics such as completion time and distance.

Our framework utilizes honey encryption (HE) [16] to provide its functionalities. The spatial trajectory data is stored in service provider's database. A user is required to authenticate herself using password to access her spatial trajectory. If the user enters the correct password, server provides the data. Additionally, our system is convenient to update distribution-transforming encoder (DTE) that encodes and decodes the message space using the specified functions. The original paper of HE [16] does not provide a solution to handle data collections that do not change over time. Thus if a new set of data will be inserted or a set of data will be deleted from the data collection, DTE must be built from scratch. However, in our case, service provider should keep DTE updated to maintain security of the proposed scheme. Therefore, we provide two protocols, one is for inserting the current DTE in the previous one and the other is for deleting a specific part of DTE, to update DTE without reconstruction.

### 3.1.3 Threat Model

The adversary in our model compromises the system using a brute-force attack, also known as a password-guessing attack. It attempts to discover a password by systematically trying every possible combination until the one correct password is discovered. In fact, although a brute-force attack guarantees discovery of the protected data at the end, it takes excessive time depending on the password's length and complexity. However, since most people choose low-entropy, easy-to-guess passwords over completely random passwords as stated in the literature [15], it is possible to speed up completion of the attack by eliminating a vast number

of passwords that are not quitely chosen by a user. For this reason, a brute-force attack cannot be considered infeasible. To put it simply, the adversary's main purpose is to break inner-layer protection and get users' spatial trajectories.

We assume that the service provider is trusted but an adversary can be any actor in the system that has access to the encrypted database. The adversary follows the protocol as specified but tries to learn more from the protocol than its role in the system is authorized, in other words, it is semi-honest (honest-but-curious). An adversary might be either an inside attacker that has access permission to the encrypted database, or an outside attacker that applies brute-force attack to get spatial trajectories on the hijacked database. Furthermore, we also consider that an adversary may have some side information about a user's whereabouts, such as public location-based check-ins.

## 3.2 Methodology

Our system based on HE is designed to realize secure storage and retrieval of spatial trajectory data by returning a plausible-looking message for each incorrect password attempt to an adversary. The steps of the protocol to store and retrieve a spatial trajectory can be seen in Figure 3.2. It is assumed that a user sends its location information to server in the form of spatial trajectory instead of periodic location updates. We utilize DTE to encode and decode spatial trajectories. As seen in Figure 3.1, after the user sends her spatial trajectory (1.1), the encoding step (1.2) is triggered. Server returns the output of encoding of spatial trajectory, seed, to client. Client performs password-based encrypted on this seed (1.4) and it stores the encrypted seed in its local storage.

When the user signals to retrieve the spatial trajectory back, client performs password-based decrypted on the encrypted seed (2.1), and sends the seed to server (2.2). Sending of the seed by client is inferred by server as a request to retrieve the corresponding spatial trajectory. Server decodes the seed (2.3) and gets the spatial trajectory as plaintext. Then, it returns the spatial trajectory to

client (2.4).

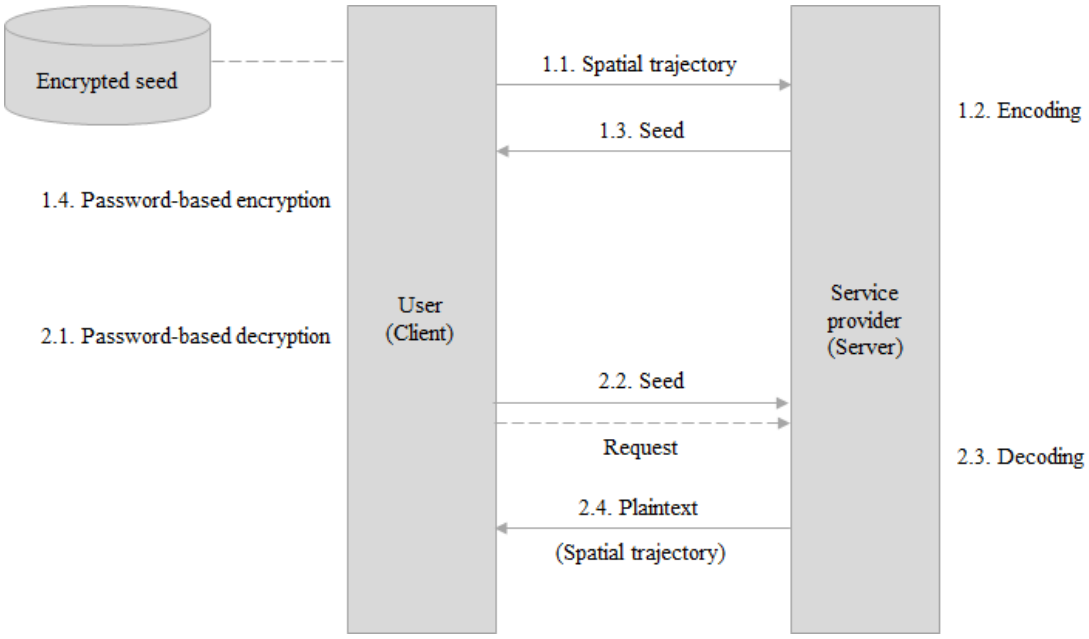


Figure 3.2: The steps of the protocol.

### 3.2.1 Preprocessing

To lay the groundwork for our novel DTE scheme, the study area is sliced into subunits over which we summarize a spatial variable for the data points lie inside as a first step. We use hexagonal tessellation, in which a grid of regular hexagonal cells is overlaid on a study area and each cell is assigned a set of values for the spatial variables of interest. To construct the DTE scheme, the variables we include are the number of data points per grid cell and the transition probabilities of data points from a cell to six neighboring cells and to itself. In the context of this application, a transition means a change of location of a sample from one cell at time $t = i$ to time $t = i + 1$. It should be noted that the elapsed time between two locations is the unit time used to sample data so that at the end of a transition a sample cannot move more than one unit Manhattan distance between two cells. The notation of $T_{(a,b)}$ shows a transition from hexagon $H_a$ to hexagon $H_b$, where $Manhattan(H_a, H_b) \leq 1$. Accordingly, $H_b$ must be one of the

seven cells including six neighbors and the cell itself.

Although using square cells is the most common method for defining a spatial grid in the literature and any regular tessellation of the plane can be chosen to apply our solution, we determine to use hexagonal grid that helps to boost the efficiency of the proposed method. There are some drawbacks of using hexagonal grid compared to traditional square grid. Nevertheless, since regular hexagons are the closest shape to a circle that can be used for the regular tessellation of a plane and the additional symmetry they have reduces the amount of space that should be allocated for the application and the computational complexity of generating the DTE scheme, we use it for this application. In Chapter 5: Discussion, the properties of hexagonal grids will be compared with square grids and their benefits will be put forth. The details of the DTE scheme and the data structure used to construct it will be given in the next section.

### 3.2.2 Encoding

In Preliminary, a spatial data structure that is used to divide the study area into uniform regions called cells using hexagonal tessellation is introduced. With the help of this structure, we propose a DTE scheme that leverages the application of HE method making enable to use for spatial trajectories. We define a spatial trajectory as a sequence of consecutive grid cells. Based on this definition, the general idea of our DTE scheme is estimating the conditional probability of the destination cell of a spatial trajectory given all preceding cells. The probability of complete sequence M can be calculated by composing the probability of each preceding cell consecutively, where $P(m_i|M_{1,i-1})$ denotes the conditional probability of the $i$-th cell given preceding ones:

$$P_m(M) = P(m_n|M_{1,n-1})P(m_{n-1}|M_{1,n-2})\dots P(m_2|m_1)P(m_1)$$

The computation of the conditional probability $P(m_{i+1}|M_{1,i})$ will be given later. In this section, we analyze the encoding process on higher level.

The approach taken to realize HE is through a DTE that consists of a randomized encoding and deterministic decoding algorithm. In constructing the HE scheme, the main challenge is efficiently encoding a message into a uniformly distributed seed. The process of encoding is simply mapping a message $M$ to the corresponding portion of seeds $S_M$, and then uniformly picking a value from $S_M$.

To efficiently encode a spatial trajectory, namely a grid cell sequence, we follow an approach in which subspaces of $S$ is assigned to the prefixes of a sequence $M$. In this context, a prefix is a subsequence of the message that starts from the origin cell of the message. For the sequence $M$, the prefixes are the subsequences in the set $M_{1,i}|1 \leq i \leq n$. Suppose $M$ is a spatial trajectory with cells $m_i|1 \leq i \leq 4$. The prefixes of $M = \{m_1, m_2, m_3, m_4\}$ are $\{m_1, m_1m_2, m_1m_2m_3, m_1m_2m_3m_4\}$.

In the setup of encoding approach, we construct a tree-based DTE to encode spatial trajectories. In this tree structure, there are branches in the tree for each spatial trajectory, and a seed subspace $S_M$ is assigned for each branch. Throughout encoding, a random seed from $S_M$ is picked for the sequence $M$ that represents a spatial trajectory. The DTE tree has n levels, where nodes at $i$-th level stand for possible cells that a spatial trajectory can be located in time interval $t = [i, i + 1)$. Moreover, each node has at most seven edges that are connected to neighbor nodes, since a hexagonal cell has six neighboring cells. To be more precise, $node_{i,j}$ represents $i$-th level and $j$-th order of the tree, and it corresponds to the state of being located in cell $m_j = m_0$, where its neighbors are $\{m_x|1 \leq x \leq 6\}$. The next cell $m_y$ can only be the cells $\{m_y|0 \leq y \leq 6\}$ at the end of unit time with any possible transition.

For the level 0, the seed space is assigned to the root node entirely. The assigned seed space for level 0 $[L_0^0, U_0^0]$ is the available seed space that will be distributed in portions for the nodes at higher levels. To represent the available seed space, we define a variable called $avail$. The available seed space of $node_{i,j}$ can be calculated as $avail_{i,j} = U_i^j - L_i^j + 1$. For each node in a level, $avail$ value of parent node is distributed into sub seed spaces in direct proportion to corresponding conditional probabilities. If $alloc$ variable is called as the allocated sub seed space to a node by its parent, the available seed space of a parent equals

the total sub seed space allocations done for the children nodes. For a node $node_{i,j}$ with $N$ number of children at level $i+1$, the total of allocated sub seed space is: $\sum_{j=1}^{N} alloc_{i+1,j} = U_i^j - L_i^j + 1$.

In brief, a sequence can be encoded using a perfect 7-ary tree, and calculation of allocated seed subspaces for each child node at each level narrows the interval of root node down to the leaf node. To encode a sequence $M$, we start from the root which points to the state of locating in the origin hexagonal cell at $t = 0$. Then, we move down to a branch according to the enumeration value of next neighboring cell in the sequence. For simplicity, we enumerate child nodes in clockwise order from 1 to 6 as follows:

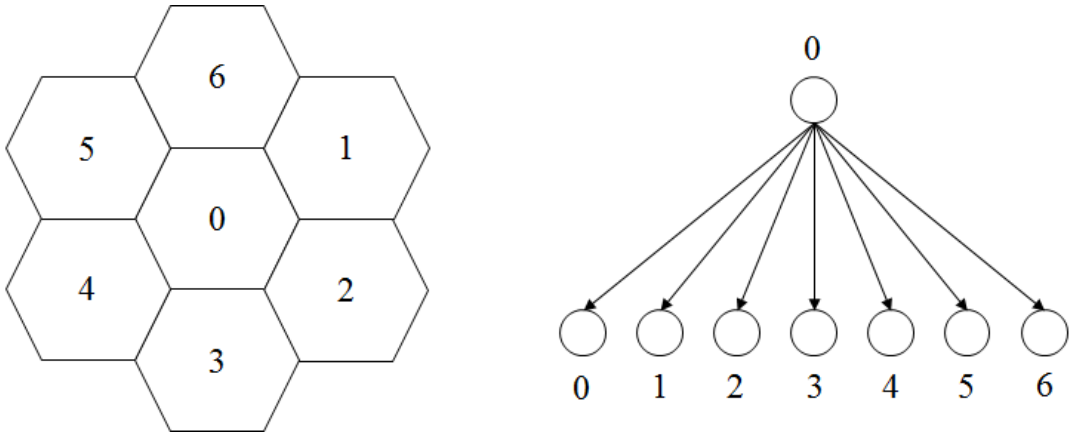


Figure 3.3: Node enumeration.

It should be noted that the cell 0 shows the current cell, and it can be the next cell in which the point that follows the origin in the sample trajectory lies in. If the next cell is 0, we should move down from the current node to the leftmost branch. If the next cell is 1, this time, we should move down from the current node to the branch where is on right of the leftmost one. The same order continues for the remaining neighbors till the 6-th one which corresponds to the rightmost branch. In this structure, each interval node represents prefixes of a sequence, and leaf node represents a complete sequence. Moreover, we attach an interval $[L_i^j, U_i^j)$ to each node using the conditional probabilities. In interval $[L_i^j, U_i^j)$, $i$ stands for the depth of the relevant node in the tree and $j$ is for the order of that node. Both values $i$ and $j$ start from 0. Also, the interval of a node

is the sub seed space which will be assigned to a sequence that starts with the prefix (or complete sequence if it is the interval of a leaf node) represented by that node.

Suppose the root has an interval $[L_0^0, U_0^0) = [0, 1)$ and we encode a sequence $M$. Encoding performs the following calculation from the node $M_{1,i}$ with order $j$ at depth $i$ to depth $i+1$ depending on the node enumeration value $m_{i+1}$ of next cell in the message.

- If $m_{i+1} = 0$, go to the leftmost branch and attach an interval:
  $[L_{i+1}^{7j}, U_{i+1}^{7j}] = [L_i^j, L_i^j + alloc_{i,7j} - 1]$.
- If $m_{i+1} = 1$, go to the branch that on the right of the leftmost one and attach an interval:
  $[L_{i+1}^{7j}, U_{i+1}^{7j}] = [L_i^j + alloc_{i,7j}, L_i^j + alloc_{i,7j} + alloc_{i,7j+1} - 1]$.
- ...
- If $m_{i+1} = 6$, go to the rightmost branch and attach an interval:
  $[L_{i+1}^{7j+6}, U_{i+1}^{7j+6}] = [L_i^j + \sum_{x=0}^{5} alloc_{i,7j+x}, U_i^j]$.

We can generalize the calculation of the interval for a child node $m_{i+1} = k$, where $N = 7$ shows the total number of edges that a node has as follows:

$$[L_{i+1}^{Nj+k}, U_{i+1}^{Nj+k}) = [L_i^j + \sum_{x=0}^{k-1} alloc_{i,Nj+x}, U_i^j)$$

The above formula is used to calculate intervals at each level. The encoding algorithm narrows down the available interval by looking these intervals based on the node enumeration value of next child in the input message. This process ends when we reach to a leaf node with the interval $[L_n^j, U_n^j)$. Eventually, we randomly select a seed in the interval of the leaf in order to encode the input sequence. In the next subsection, this encoding process will be exemplified.

### 3.2.3 Encoding Example

For this toy example, assume that all sequences are of length 3. The order of node enumeration of the sequence $M$ that will be encoded is $(0, 5, 2)$. Assume the seed space $\mathcal{S}$ includes the seeds in the range of $[0, 1000)$. The illustration of DTE that is used for encoding can be seen in Figure 3.4. This figure does not illustrate all branches entirely but includes the nodes that we need to perform encoding of the given sequence in order to simplify the presentation of the DTE tree. Assume $P(m_1 = 0) = 0.2$, $P(m_2 = 5|m_1 = 0) = 0.1$, and $p(m_3 = 2|M_{1,2}) = 0.3$. According to these transition probabilities, the encoding can be performed as follows:

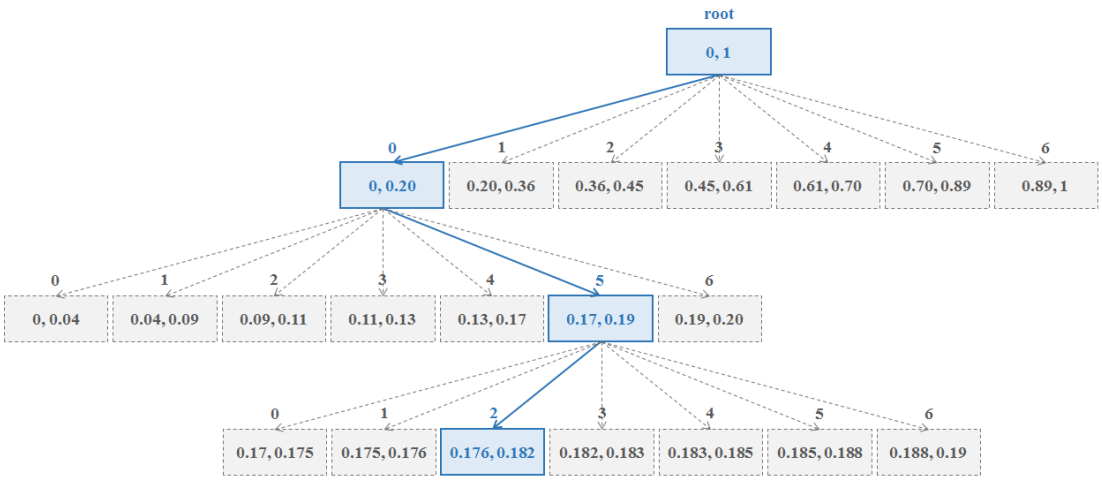


Figure 3.4: A toy example of encoding process.

- We know that $P(m_1 = 0) = 0.2$, so 20% of the root interval is $[0, 200)$.

The intervals for the first three children of next interval are:

- $[L_2^0, U_2^0) = [L_1^0, L_1^0 + (U_1^0 - L_1^0) \times P(m_2 = 0|m = 1 = 0)) = [0, 0.04)$
- $[L_2^1, U_2^1) = [L_1^0 + (U_1^0 - L_1^0) \times P(m_2 = 0|m = 1 = 0), L_1^0 + (U_1^0 - L_1^0) \times (P(m_2 = 0|m_1 = 0) + P(m_2 = 1|m_1 = 0))) = [0.04, 0.09)$

- $[L_2^2, U_2^2) = [L_1^0 + (U_1^0 - L_1^0) \times (P(m_2 = 0|m_1 = 0) + P(m_2 = 1|m_1 = 0)), L_1^0 + (U_1^0 - L_1^0) \times (P(m_2 = 0|m_1 = 0) + P(m_2 = 1|m_1 = 0) + P(m_2 = 2|m_1 = 0))) = [0.09, 0.11)$

Using this pattern, the following function is derived to calculate the interval for a child:

$$[L_j^i, U_j^i) = [L_{j-1}^i + (U_{j-1}^i - L_{j-1}^i \times \sum_{t=0}^{i-1} P(m_i = t|m_{i-1})), L_{j-1}^i + (U_{j-1}^i - L_{j-1}^i \times \sum_{t=0}^{i} P(m_i = t|m_{i-1})))$$

Applying this, we can calculate $[L_2^5, U_2^5)$ as $[0.17, 0.19)$, and $[L_3^2, U_3^2)$ as $[0.176, 0.182)$. It should be noted that we did not need to compute all of the intervals seen in Figure 3.4 when encoding the sequence $(0, 5, 2)$. We only make calculation for the intervals in blue line. After we reach the leaf $[0.176, 0.182)$, we pick a random number in this range, e.g., 0.177. Since our seed space is $[0, 1000)$, the randomly picked number maps to the seed 176.

After the seed is picked and the encoding is finished, the plain seed is given to a conventional password-based encryption (PBE). Using the provided user password, the plain seed is encrypted. Then, the encrypted seed is sent to the service provider's centralized database. If a user makes a request to get her spatial trajectory, the system responds with the encrypted seed. In order to return the sequence to the user, the encrypted seed is decrypted, and the output seed is decoded. In the following, the decoding process in which the plain seed is transformed to the sequence will be explained.

### 3.2.4 Decoding

The decoding algorithm is used in decryption to output the message $M$ getting the corresponding seed $S$, where $S \in \mathcal{S}$ and $M \in \mathcal{M}$. Before decoding, a request to retrieve the trajectory data is sent by the user. With this signal, the service

sends the encrypted seed $C$. Using the user password, the encrypted seed $C$ is decrypted and the plain seed $S$ is retrieved. At last, the seed $S$ is decoded and the message sequence $M$ is generated.

The process of decoding reverses what is done in encoding algorithm, and its way of calculation is similar. Likewise encoding, the underlying machine of decoding starts from the root of the DTE tree. Then, at each level the intervals are calculated using the conditional probabilities down to the last level. The encoding is a randomized algorithm in which a degree of randomness as a part of its logic is employed using. To guide this behavior, the encoding algorithm uses a random integer as an auxiliary input. On the other hand, the decoding algorithm is deterministic, and for this reason it always produces the same output. While passing through a level of the tree, the decoding algorithm performs comparison of the given seed with the interval of each node in the level that is in progress. If the node with the interval that includes the given seed is found, that node is chosen to narrow the current interval down. The same process is repeated until the last level. When a leaf node is found in the last level, the sequence of nodes from the root to the leaf is returned as output.

### 3.2.5 DTE Tree Update Functions

Our model has the Markov property in which the conditional probability distribution for the system at the next step depends only on the current state of the system. However, this model does not provide maintainability for the system. The previous techniques are only designed to realize secure storage and retrieval of spatial trajectories but they could not be used to update stored seeds when a change is performed on the DTE tree.

For a given spatial trajectory, we can follow branches of succeeding states according to the order of the sequence of cells starting from the root node of the corresponding DTE tree. When we reach the leaf node that stands for the last state, we acquire the proper seed range to encode the target trajectory. From the root to the leaf, a DTE tree is able to represent transitions that are done in a

specific time interval $D_i$. If we want to maintain our DTE tree for the transitions that are done in the next time intervals, we have to reconstruct the DTE tree from scratch. Likewise, if we want to modify the tree to represent a sub-interval $Dsub_i \in D_i$, we have to reconstruct it again. Nevertheless, this naive approach is inapplicable in a real-life scenario. If the original DTE tree has many levels and we want to realize insertion or deletion on that tree, the naive method does not work in reasonable time due to the intolerable complexity of reconstruction. In order to bridge this gap in our system, we propose two techniques; DTE tree insertion and DTE tree deletion.

#### 3.2.5.1 DTE Tree Insertion

In our model, we assume that the DTE tree is updated on regular basis. At the end of each updating period, a new DTE tree that holds probabilities of transitions performed in that period is constructed. Then, using the new DTE tree, the old seed is modified. This operation assists to eliminate one of the major limitation of HE, constructing a DTE for dynamic datasets.

The insertion protocol is given in Figure 3.5. The protocol starts with construction of new DTE tree by server (1). Using this tree, the seed is retrieved and it is sent to client (2). Client decrypts old seed (3), and calculates new seed using the old seed and the partially new seed sent by server (4).

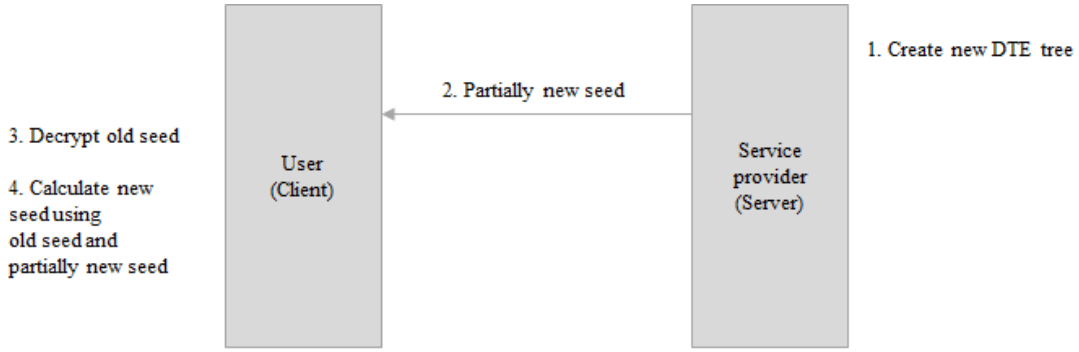


Figure 3.5: Insertion protocol.

Assume $L_{old}$ and $U_{old} = L_{old} + a$ are lower bound and upper bound of the interval that corresponds to seed respectively. Likewise, $L_{pnew}$ and $U_{pnew} = L_{pnew} + b$ are lower and upper bounds of the interval of partially new seed. The interval of new seed $[L_{new}, U_{new})$ can be computed as follows:

$$[L_{new}, U_{new}) = [a \times L_{pnew} + L_{old}, a \times U_{pnew} + L_{old})$$

#### 3.2.5.2 DTE Tree Deletion

In addition to insertion operation, deletion operation can be performed as well on DTE tree to provide maintainability for dynamic spatial trajectory datasets. The insertion operation is simply constructing a new DTE tree, and updating seed information at client. In order to support deletion operation for our model, we consider that after each insert, client does not only hold the new seed but also it continues to hold old seed as well. Thence, client holds a seed for each inserting iteration.

Figure 3.6 explains the deletion protocol. The deletion protocol starts when server indicates deletion point to client (1). Deletion point $X$ is simply an integer that shows which seed will be used as the target by client. Then, client decrypts seed that is holded for time $X$ (2). The intervals of this seed is shown as $[L_X, U_X)$. Client also decrypts the last seed that is stored (3). Intervals of last seed $N$ is shown as $[L_N, U_N) = [L_X + a, L_X + a + b)$. At last, client computes the new seed (4).

The formula to compute new seed for deletion is as follows:

$$[L_{new}, U_{new}) = [\frac{a}{U_X - L_X}, \frac{a+b}{U_X - L_X})$$

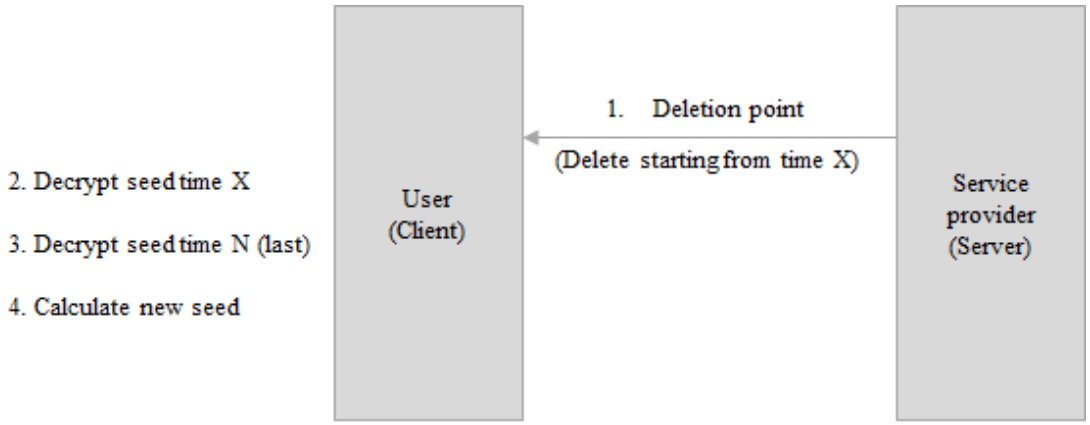


Figure 3.6: Deletion protocol.

# Chapter 4

# Security Analysis

This chapter contains the security analysis of proposed DTE model in Chapter 3, and it evaluates the security guarantee of the system against different types of brute-force attacks. We will analyze these attacks dividing into two main groups; (i) the attacks in which an adversary has only the public information (the location of cells corresponds root and leaf nodes) and (ii) the attacks in which an adversary has background knowledge about some visited location points in the spatial trajectory that it tries to reveal.

## 4.1 Measure for DTE Security

The encoding algorithm allocates a seed space of size $7^{n-i-1}$ to a branch at step $i$. At each following step, an input interval is segmented into seven parts of equal size. This allocation accordingly guarantees that each sequence in the sub-tree under the branch of step $i$ matches up with only one seed. The subinterval of the $j$-th node at depth $i$ of the tree contains $7^{n-i-1}$ integers which is the exact number of sequences under the relevant branch. The DTE construction enables transformation of non-uniform distribution of messages to a uniform space. Hence, decoding uniform points in the seed space provides a sampling close to that of

the target distribution $p_m$. For a seed space $\mathcal{S} = [0, 2^{hn} - 1]$, the DTE message distribution over $\mathcal{M}$, $p_d$, is defined below:

$$p_d(M) = P[M' = M : S \leftarrow_{\$} \mathcal{S}; M' \leftarrow decode(S)]$$

The security of a HE scheme depends on the difference between $p_m$ and $p_d$ distributions. A DTE is secure as much as the two distributions, $p_m$ and $p_d$, are close to each other. At this point, the quantification of this difference for the proposed DTE scheme is given. The original probability of the prefix sequence $M_{1,i}$, $P_m^i$, is, $P_m^i = \sum_{M' \in \mathcal{M}, M'_{1,i} = M_{1,i}} = p_m(M')$. In the same way, we define $P_d^i$ in the distribution $p_d$. The detailed proofs and analysis are given in [10].

**Lemma 1.** $\forall M \in \mathcal{M}, |p_m(M) - p_d(M)| < \frac{1}{2(h - log_n^7)n}$. Lemma 1 bounds the largest difference between $p_m(M)$ and $p_d(M)$. It leads us to the HE theorem that bounds the DTE advantage of an adversary. The following definition gives the DTE advantage.

| $\underline{SAMP1^{\mathcal{A}}_{DTE}}$ | $\underline{SAMP0^{\mathcal{A}}_{DTE}}$ |
|---|---|
| $M^* \xleftarrow{p_m} \mathcal{M}$ | $S^* \xleftarrow{\$} \mathcal{S}$ |
| $S^* \xleftarrow{\$} encode(M^*)$ | $M^* \leftarrow decode(S^*)$ |
| $b \xleftarrow{\$} \mathcal{A}(M^*, S^*)$ | $b \xleftarrow{\$} \mathcal{A}(M^*, S^*)$ |
| return $b$ | return $b$ |

Figure 4.1: Game in which the DTE advantage is defined. In $SAMP1^{\mathcal{A}}_{DTE}$, sequence $M^*$ is sampled according to $p_m$, whereas in $SAMP0^{\mathcal{A}}_{DTE}$, $M^*$ is equivalently sampled according to the DTE message distribution $p_d$. The output $b$ that can be returned to adversary should be either 0 or 1. These two values indicates the guess of adversary if it is in $SAMP0^{\mathcal{A}}_{DTE}$ $(b = 0)$ or $SAMP1^{\mathcal{A}}_{DTE}$ $(b = 1)$.

**Definition 1.** Let $\mathcal{A}$ be an adversary who attempts to distinguish the two games which are given in Figure 4.1. The advantage of $\mathcal{A}$ for the original message distribution $p_m$ and encoding scheme $DTE$ is

$$Adv^{dte}_{DTE,p_m}(\mathcal{A}) = |P[SAMP1^{\mathcal{A}}_{DTE} \Rightarrow 1] - P[SAMP0^{\mathcal{A}}_{DTE} \Rightarrow 1]|$$

**Theorem 1.** Let $p_m$ be the original message distribution and $DTE$ be the distribution-transformation encoder scheme which realizes encoding and decoding using $hn$-bit. Let $\mathcal{A}$ be an adversary, then

$$Adv^{dte}_{DTE,p_m}(\mathcal{A}) \leq \frac{1}{2^{(h-2\ log_2^7)n}}$$

The proof follows Theorem 6 in [34].

### Message Recovery Security

The securtity analysis is concluded with quantification of the message recovery (MR) security which is provided for the encryption scheme $HE$ against any adversary $\mathcal{B}$.

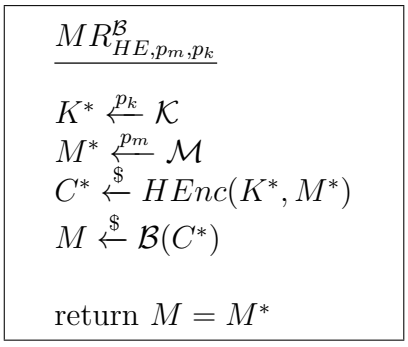


Figure 4.2: Game in which the MR security is defined. Let $C^*$ be the ciphertext that is encrypted from $M^*$ and let $\mathcal{B}$ be the adversary that is permitted to reveal the message by performing brute-force attack. The game is won by $\mathcal{B}$ if the original message $M^*$ is same with the output message $M$.

**Definition 2.** Let $\mathcal{B}$ be the adversary that attempts to reveal the correct sequence given honey encryption of the sequence, as shown in Figure 4.2. The advantage of the adversary $\mathcal{B}$ against the encryption scheme $HE$ is

$$Adv^{mr}_{HE,p_m,p_k}(\mathcal{B}) = P[MR^{\mathcal{B}}_{HE,p_m,p_k} \Rightarrow true]$$

It should be noted that herein the password distribution $p_k$ is non-uniform. It is assumed that $w$ shows the probability of the password with the highest probability. With the help of Lemma 1 and Theorem 1, the below theorem is defined.

**Theorem 2.** Considering that $HE[DTE, H]$ with $H$ (the hash function) is modeled as a random oracle and encoding scheme $DTE$ using $hn$-bit. Let $p_m$ be the original message distribution with maximum sequence probability $\gamma$, and $p_k$ be a password (key) distribution with maximum weight $w$. Let $\alpha = \lceil 1/w \rceil$. Eveuntually, for any adversary $\mathcal{B}$,

$$Adv^{mr}_{HE,p_m,p_k}(\mathcal{B}) \leq w(1+\delta) + \frac{7^n + \alpha}{2^{(h-\log_2^7)n}},$$

*where* $\delta = \frac{\bar{\alpha}^2}{2\bar{b}} + \frac{e\bar{a}^4}{7^7\bar{b}^2}(1 - \frac{e\bar{a}^2}{\bar{b}^2})^{-1}$ *and* $\bar{a} = \lceil 7/w \rceil$*and* $\bar{b} = \lfloor 2/\gamma \rfloor$

The proof is similar with Corollary 1 in [34]. The detailed information about Theorem 1 and Theorem 2 can be found in [34, 10].

## 4.2 Security under Brute-Force Attacks

In order to evaluate security of the proposed system, this paper presents two exploratory experiments to compare a conventional PBE algorithm with the aforementioned approach under brute-force attacks. Brute-force attacks are applicable in computationally reasonable time only if the adversary knows that a valid password has a limited number of characters. For these experiments, we supposed that the adversary has known the password pool in which passwords are integers from 0 to 999. We encrypted a randomly chosen spatial trajectory under a given password from the pool. We associated this sequence with the password "548", and use it as the correct password for both experiments. Then, we implemented a brute-force attack to reveal the correct sequence by trying all possible passwords.

In the first experiment, a simple PBE algorithm [17] is used to encrypt the spatial trajectory after encoding by assuming uniform probability distribution of transitions. In other words, we set all probabilities equal (1/7, since each node has 7 edges) for all edges in the tree. In this case, the adversary will obtain a valid but not plausible-looking spatial trajectory when it decrypts the ciphertext under an incorrect key. In the second experiment, the same setting is used but

this time instead of the PBE scheme, the encryption of the sequence is performed by using our proposed approach.

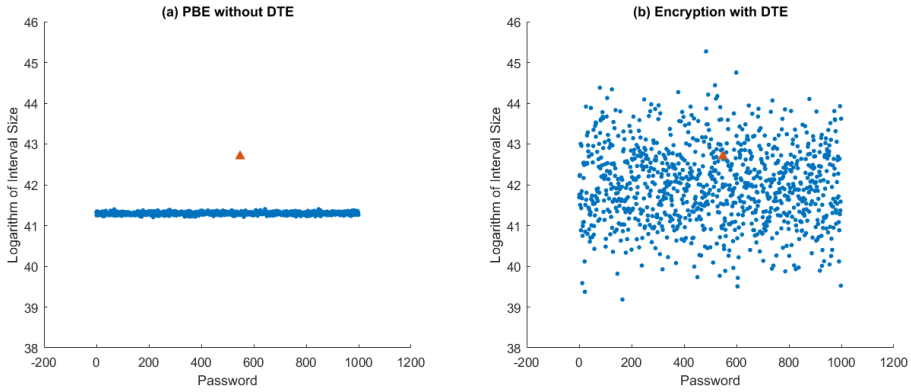


Figure 4.3: Security evaluation: Comparison of a simple brute-force attack on a conventional PBE and on the proposed system

The size of the interval of a leaf in the tree is proportional to the probability of the corresponding sequence in our proposed DTE scheme. On the basis of this observation, we can rule out wrong passwords comparing the computed interval sizes of the decrypted sequences. In Figure 4.3, the results of the two experiments are given in plots where left side represents the first experiment and the right side represents the second one. In both plots, each point represents one decryption result using an integer from the password pool. As seen in Figure 4.3 (a), the corresponding point for the correct sequence stands apart from the other points. The reason is that the corresponding decrypted sequences have considerably lower probabilities than the probability of the correct sequence. Thus an adversary will be able to exclude the wrong password, and identify the correct sequence using a simple classifier. On the contrary, in Figure 4.3 (b), which illustrates the results if our approach is used for the encryption, the point that corresponds to the correct sequence blends in with the other points. For this reason, it is nearly impossible for an adversary to exploit the difference between the probability of correct sequence and the probabilities of other sequences.

To sum up, the two experiments show us that our approach significantly prejudice the chance of the adversary to classify the sequences with low probability

and to reveal a correct sequence by excluding the ones encrypted under wrong passwords. Nevertheless, the same security guarantee cannot be provided if the adversary has some side information about a sample's spatial trajectory. In the next section, we will examine our system in several settings in which we assume that the adversary has known different amounts of places in the target sequence.

## 4.3 Security Against Adversaries with Side Information

An adversary can obtain information about some point of interest visited by a specific LBS user, when that user actively check ins to let public know about its whereabout as a matter of course. Moreover, if a user had agreed for their location to be shared with others, that user's contacts would be able to easily share their location publicly on web. An adversary may craft a script that collects the check-in information, and use this side information to expose additional sensitive information.

The mentioned privacy issue risks our proposed system as well. Some point of interests visited by a user can be known by the adversary. This side information may help the adversary to find out the spatial trajectory that belongs to the target user entirely. In order to analyze the security of our system against an adversary with side information, we assume a set of possible point of interests (hexagonal cells in our context) such as $H_1, H_2, \ldots, H_u$. We assign these possible cells in which the target user is potentially located by counting the number of transition done through them. The most visited five cells $H_{P_1}, H_{P_2}, H_{P_3}, H_{P_4}, H_{P_5}$ are selected as the possible point of interests for our analysis. The potential adversary in our experiment knows whether a target spatial trajectory includes a cell in the set and which cell is it. Suppose the adversary with the side information performs a brute-force attack. It tries each password in the pool and decrypts the ciphertext. As a result of the decryption, the adversary obtains a sequence. Then, it authenticates this sequence using its side information. The tried password is

kept or excluded with a binary decision. If the result sequence matches with the acquired side information, the tried password is kept. Otherwise, the password is excluded from the pool.

Suppose that there are $N$ passwords in the pool before the adversary starts to use classifier to rule passwords out. These passwords are listed in descending order regarding their probabilities: $P_1 \geq P_2 \geq \ldots \geq P_N$, where $\sum_{i=1}^{N} P_i = 1$. The aforementioned order of a password is commonly called the rank of that password. In the literature, it has been shown that the distribution of real-life passwords follows Zipf's law [35, 36]. The Zipf's law asserts that there is an inverse proportion between the frequency of any word and its rank in the frequency table for a given corpus of natural language iteration. Thus, the probability of password with rank $i$ is the following:

$$P_i = W i^{-r}$$

where $W$ and $s$ are constants depending on the dataset.

Suppose $H^*$ is a point of interest (hexagonal cell) that is known to the adversary. Decryption under an incorrect key results a spatial trajectory that includes $H_i$ with the probability $P_{H_i}$. It should be noted that this assignment is independent across passwords. We can compute the probability of retaining a password $P_{ret}$ as independent Bernoulli trials across passwords as follows:

$$P_{ret} = \sum_{i=1}^{N} P_{H_i}$$

As Theorem 2 states, the advantage of adversary $\mathcal{B}$ without side information is approximately equal to $w$ (the maximum weight of a password in the password distribution). Suppose $\mathcal{B}'$ is the adversary with side information $H^*$. Initially, adversary $\mathcal{B}'$ prunes passwords applying the classifier. Then, it applies the algorithm of adversary $\mathcal{B}$ in the message recovery game on the pool of retained passwords. Let $p'_k$ be the new password distribution with maximum weight $w'$.

We realize the randomization of the password pruning process using the function, $f(p_k) \rightarrow p'_k$. Hence, adversary $\mathcal{B}'$ uses randomized function $f$ to compute $p'_k$, then $\mathcal{B}'$ gives $p'_k$ to $\mathcal{B}$. We can calculate the advantage of $\mathcal{B}'$, $Adv(\mathcal{B}')$, as follows:

$$Adv(\mathcal{B}') = E_{p'_k \leftarrow f(p_k)}[Adv^{mr}_{HE,p_m,p'_k(\mathcal{B})}] \approx E_{p'_k \leftarrow f(p_k)}[w']$$

where $E$ is the expectation over the randomized password pruning process, and we approximate $Adv^{mr}_{HE,p_m,p'_k(\mathcal{B})}$ with the maximum weight $w'$ in the password distribution $p'_k$. In order to quantify $Adv(\mathcal{B}')$ using real data, we can make use of the Ziph's model for password distribution [35]. We use the following settings that is given in [35]: $t = 486118$, $W = 0.037871$ and $r = 0.905773$.

We evaluate the privacy loss by considering each point of interest as known side information for adversary $\mathcal{B}'$. In our experiment, we perform the Bernoulli trials with corresponding $P_{ret}$ on the password pool. Then, we calculate $Adv(\mathcal{B}')$. The evaluation of adversary's advantage with specific point of interests as the side information is shown in Figure 4.4:

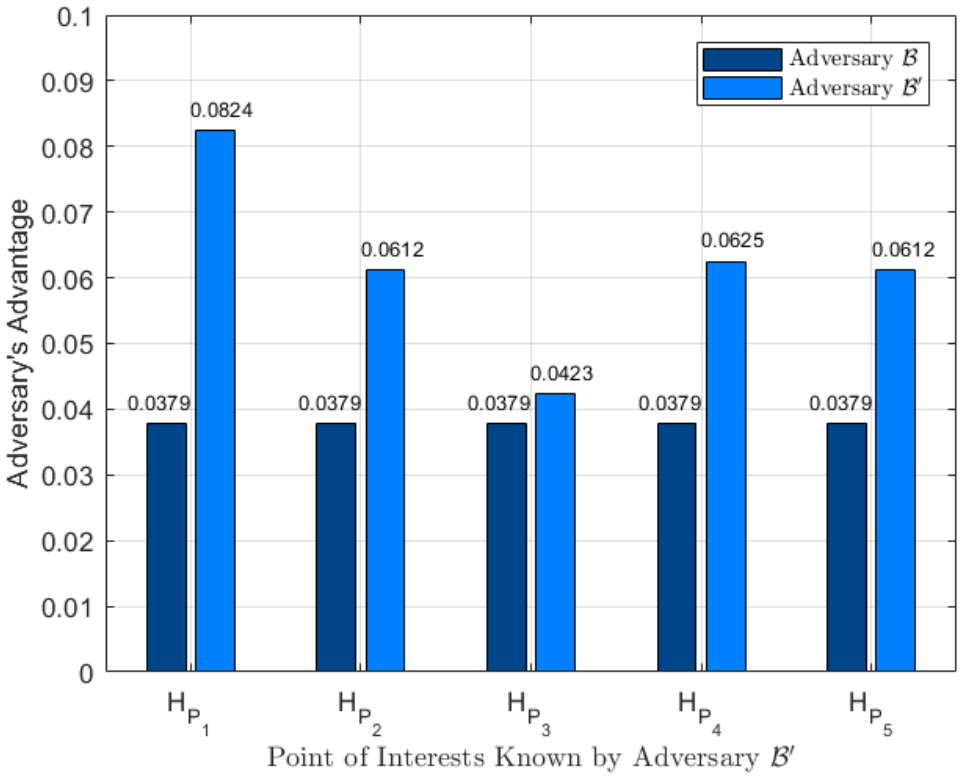


Figure 4.4: Users' point of interests known by Adversary $\mathcal{B}'$

# Chapter 5

# Discussion

In this section, we analyze the performance of the proposed method and put forth its application areas. We also mention some implementation details and discuss about limitations of the scheme.

## 5.1 Performance

The encoding and decoding algorithms run in O($N$) where $N$ is the length of the sequence. We analyze the performance of retrieval process including four main steps: *encode*, *decode*, $PBE\_encrypt$, $PBE\_decrypt$.

We implemented the system in Python (encryption, decryption) and Java (calculation of transition probabilities on hexagonal tessellation), and ran it on Intel(R) Core(TM) i7-4720HQ CPU @ 2.60GHz 64-bit Windows Operating System.

The password-based encryption and decryption of the implementation conforms the password-based cryptography standard, PKCS #5 [17]. Accordingly, HMAC-SHA-1 is used as the underlying pseudorandom function. For a given password $P$ and a 64-bit random salt $R$, we applied a key derivation function $KDF$ to get a 128-bit derived key $DK$:

$$DK = KDF(P, R)$$

The derived key DK is used as the key for an AES block cipher that encrypts the seed in CBC mode. We used CRAWDAD EPFL/Mobility Dataset that contains mobility traces of 500 taxi cabs collected over 30 days in San Francisco Bay Area, USA [37]. We evaluated the system on 250 spatial trajectory samples dividing them into 5 clusters of different sizes. The average performance for each cluster of 50 samples is shown in Figure 5.1.

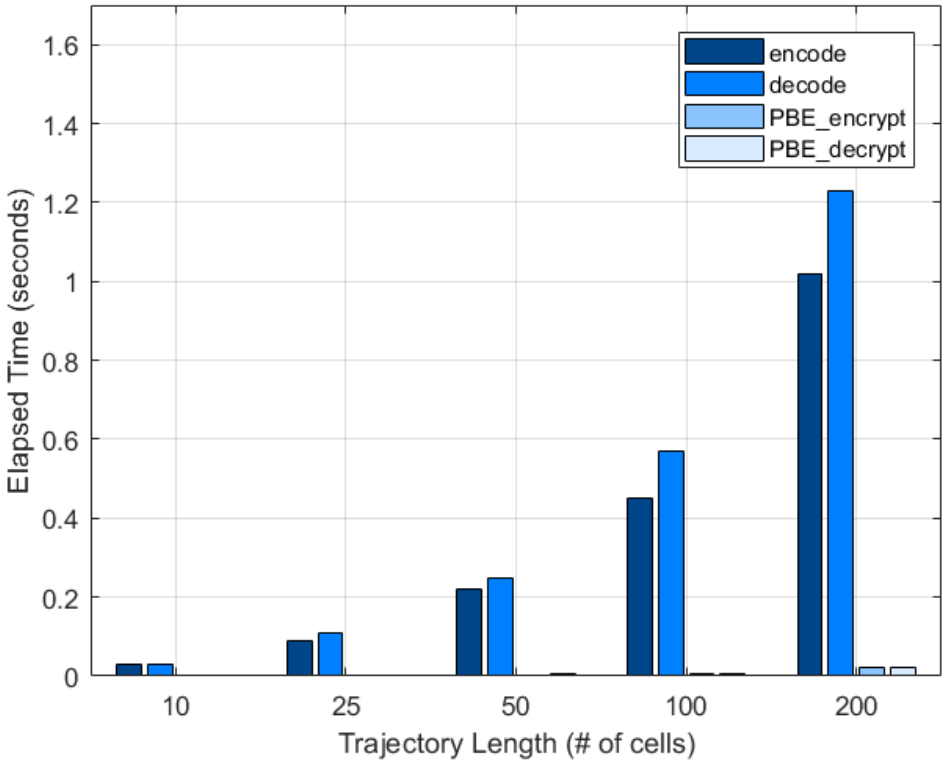


Figure 5.1: Performance evaluation of the model.

As can be seen in the figure, encoding and decoding processes are the determining factor for the running time of overall retrieval process. The running time of both encoding and decoding shows approximately linear dependence with the length of a spatial trajectory. The elapsed time to perform password-based

encryption and decryption is negligible comparing encoding and decoding.

## 5.2 Application Areas

The LBS providers accumulate location samples from the service consumers who gave permission in mobile applications. In return, customized information for consumers' whereabouts are presented. The vast majority of LBS are not only based on users' current locations but also previous location reports. Although users of local search-and-discovery service mobile applications manually share their locations via self-reported positioning whenever they open and use/interact with those applications, their locations are implicitly listened in background if users have turned on the relevant functionality, and stored in spatial databases in order to improve user experience even they are not directly interacting with the application [38]. In other respects, there are applications which require users' location continuously to offer several services such as road-traffic monitoring, social networking, fitness tracking, family member tracking, and pet tracking. In general, periodic location updates are stored in the form of spatial trajectory. Then, by exploiting the spatial trajectory data, users connect with other users based on commonality of places they frequent, discover new places near their previous routes, display the history of traces, etc. The categories of LBS that can use our technique to secure their databases against brute-force attacks are as follows.

**Maps and Navigation** Thanks to advances in web mapping services (e.g., Google Maps), finding a route between two locations become one of the most popular use of LBS. Typically, the user indicates its location to an LBS and makes a request for the route to another location. In this use case, users have to reveal their current location and a potential destination. Unfortunately, adversaries that have access control to databases that hold these data, may obtain statistical information about these locations and find home and work location of users [38]. The web mapping services provide route planning for traveling by

foot, car, bicycle, or public transportation. The services like Google Traffic provide real-time traffic conditions to their users. In order to increase the quality of their services, majority of these kind of LBS listen users' locations' periodically.

**Location Check-ins** Location check-ins, also known as self-reported positioning, is a low cost way to using location technology to know users' whereabouts comparing to tracking location continuously. Users check in to let their friends know where they are, what kind of activities at that time are they do, and how their experiences about the visited location are. In this way, users are used as source of information about locations and in return LBS give access for information provided by other users related to the visited location. Together with its benefits, these services may be abused by adversaries, since users reveal their location and intention to be located there. If an adversary is able to access the types of places that a user is located before, the privacy of that user may be violated, since not only the user's samples of mobility will be seen but also the adversary can distinguish between the user's check-ins in public places and its private home [38].

**Localized Search** Localized searches offer mobile users access to display local services around a target location. Depending on the type of information provided by the LBS, users can obtain location of educational institutions, medical centers, restaurants or other venues [38]. Although this type of services is useful for users while discovering unknown places, users' spatial trajectories can be deduced by exploiting statistical information about samples of mobility that they revealed to the LBS. The history of spatial queries made by a user during localized searches gives adversaries the chance of learning visited locations and user habits. Moreover, search results can be abused by adversaries to predict in which point of interest the user will be located after its location at that time.

**Tracking** GPS tracking solutions have application in several industries. Today a high number of people appeal to this kind of software. To exemplify, Life360 is a LBS that allows individuals to share their location with their friends

or family members. As a couple of use cases, parents can use it to locate their children if they become lost or nursing home workers and private caretakers to maintain constant contact with an elderly adult prone to wandering or getting lost. There are also some other solutions which offer a variety of device options and customizations for their users. Delivery companies, for instance, use GPS systems to parcel tracking to ensure timely delivery or overall efficiency or food and beverage services use them to keep track of deliveries and food trucks ensure the security of the product and increases the efficiency of supply delivery. GPS tracking solutions have also application area in government technologies to support security, emergency, infrastructure, and law enforcement.

Despite its widespread use cases in industry, LBS that offer GPS tracking put users' privacy at risk much more than other LBS based on manual location sharing. The underlying reason of this issue is about the architecture of these LBS. Most of GPS tracking LBS use client/server architecture instead of peer-to-peer, since they store history of users' spatial trajectory for further consideration in their databases. The spatial trajectories are used for both practical purposes like displaying previous routes of the people who are tracked and analytical purposes to improve quality of some other services. Potential attacks to service providers' databases may reveal users' spatial trajectories with exceedingly high sampling rate to adversaries, and thus users' privacy is hanged by a serious threat.

In summary, LBS sample a user's location at different frequencies and intervals depending on their functions. A number of LBS operates with manual location sharing, yet there are also respectable amount of LBS which operate with continuous spatial tracking. Since the correlation between risk and number of leaked location samples is obvious, databases that record spatial trajectories have to be protected against all potential external and internal threats.

## 5.3 Advantages of Hexagonal Geometry

This section explains why hexagonal tessellation is preferred over other alternatives and gives a analysis of mathematical properties and aspects of regular hexagons.

A regular hexagon is a convex polygon with six equal-length sides. Its each internal angle is 120 degree. It is one of three regular polygons that can be used to tessellate the Euclidean plane. The other two polygons are triangle and square. The tessellation of hexagons, nevertheless, is different than other two Euclidean tesselations, since it is combinatorially identical to the close packing of circles on a plane. For a hexagon, the distance from its center point to each vertex is equal to the length of a side.

The tessellation of hexagons on Euclidean plane in a regular pattern makes each hexagonal cell in the grid be bordered by six neighbor hexagons with no empty space left over in any direction. The hexagon is the highest-sided tessellable regular polygon. This property gives the advantage of spacing out each constituent hexagon more or less evenly from its neighbors. To put it in a different way, any given point inside a hexagon is closer to the center point than any given point in an equal-area triangle and square could be. It is obvious that triangles and squares have more acute angles comparing to hexagons. Thus, points near the corners of triangles and squares locate in further positions from the other points which locates elsewhere in the area of these shape. On the other hand, these points would locate at similar positions in a hexagon, since it has higher side number than triangles and squares. It is possible to apply this logic for other polygons with higher side number. The ultimate efficiency for enclosure of a shape area is given by circles but they do not have the tessellation advantage like hexagons. In this context, our task does not merely requires regular grid but also it requires efficient parcelling of the area by the cells of the grid. Considering these, a hexagonal tessellation is an inevitable choice for our model.

Furthermore, in hexagonal tessellation, a hexagonal cell shares only edges with

its neighbor cells, it does not share vertex borders like tessellation of triangles and squares. Triangular tessellation has nine diagonal connections (three diagonal connection for each vertex) and square tessellation has four diagonal connections with neighbor cells at its vertices.

In tessellation of squares, there is one neighbor square cell on each edge and one neighbor square cell diagonally on each vertex for every square cell. Therefore, a square cell is enclosed by eight neighbor square cells in total. In tessellation of triangles, there are one triangle cell on each edge and three triangle cells diagonally on each vertex for every triangle cell. Therefore, a triangle cell is enclosed by twelve neighbor triangle cells in total. In our encoding model that is based on DTE tree, the determining factor of complexity is the number of neighbors of a node. It means less number of neighbors in tessellation provides more efficiency. Hence, hexagonal tessellation is preferred over square and triangular tessellation in our model.

## 5.4 Cell Size Determination in Hexagonal Tessellation

The hexagonal tessellation of study area is a fundamental process to apply the proposed technique. The study area $R$ is defined using lower and upper bounds of location points of spatial trajectories. Assume $p_0$, $p_1$, $p_2$, and $p_3$ are corner points of $R$. The point $p_0$ is left bottom corner and succeeding points correspond to next corners in counterclockwise direction. Then, this area is surrounded by the grid so that each corner hexagon covers one corner point of $R$. Figure 5.2 shows an example grid that surrounds a rectangular study area.

Cell size determination is the issue of choosing the size of a cell in hexagonal tessellation. We define cell size using the maximal diameter of the unit hexagon. This variable corresponds to the long diagonal of the hexagon which is twice the maximal radius and also twice the side length.

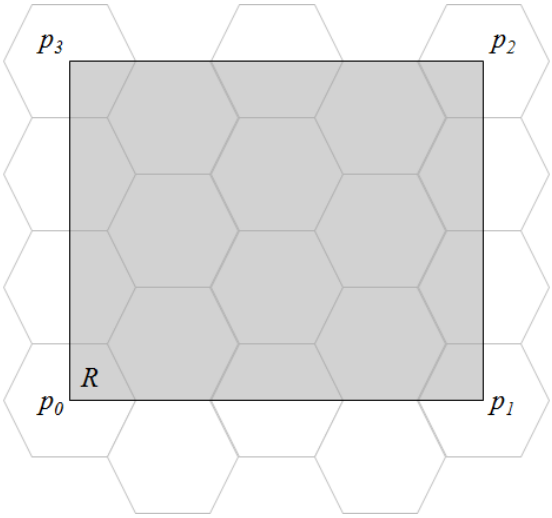


Figure 5.2: Hexagonal tessellation of study area.

The cell size directly affects the encoding and decoding mechanism of the model, since we reach a different seed by following different set of nodes on DTE tree when the cell size in the same study area $R$ is altered. In Figure 5.3, a sequence with length 100 is taken as experimental subject to examine the effect of maximal diameter on spatial variability of trajectories. In this sequence of location points, the data is sampled every 15 seconds throughout the process of movement of the chosen subject. The average spatial displacement between any sampled location point to the next one is approximately 280 meters, and the highest spatial displacement is 420 meters. We analyzed number of visited unique cells by this spatial trajectory for different cell size values.

As the cell size increases, the number of unique cells that are visited by the subject decreases down to one. It means that a serious information loss due to the over-discretization of the attributes, i.e., latitude and longitude, of location-updates will be inevitable. We came to a conclusion that in order to eliminate this information loss, the maximal diameter should be set to a value between the average and the highest spatial displacement between consecutive data point samples of a sequence. In the interval of average and highest displacement of a specific sequence, it is possible to represent the spatial trajectory without suffering from discretization.

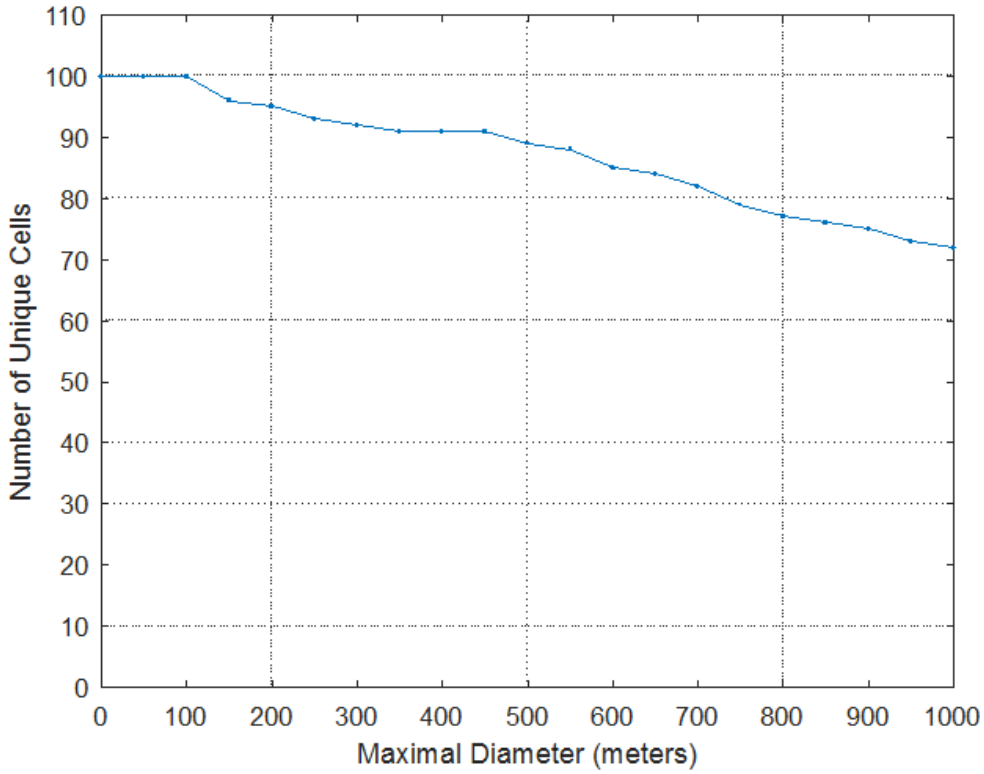


Figure 5.3: Analysis of unique cell count in spatial trajectories represented with different cell sizes.

On the other hand, as the cell size decreases, the number of unique cells that are visited by the subject increases up to the number of data points in the sequence. Although visiting as many unique cells as possible is a desired attribute while representing a sequence of location points as spatial trajectory, it may cause discontinuities in encoding of a spatial trajectory when cell size is chosen too small. The data points in this time cannot be mapped on the grid and will not be the part of the spatial trajectory. That spatial trajectory will be incomplete due to discontinuity in the location sequence. In our context, discontinuity in a spatial trajectory can be defined as an unexpected jump from a cell to the next one. In our model, we assume that if $H_i$ is a cell that the location-update in the spatial trajectory is located in time $t = i$, $H_{i+1}$ can only be one of the neighbors of $H_i$.

We eliminate the issue of discontinuity while implementing the system if we set the maximal diameter of hexagonal cells to the highest displacement between consecutive samples again. However, while constructing DTE tree, we need to consider highest displacement value for all sequences in the dataset, not only the specific one. Therefore, we calculate maximal displacement $\Delta d$ values of all sequences at first. Then, we define an upper bound $d$ for the maximal displacement. We mark a sequence $s$ as valid if $\Delta d_s < d$. Else, we segmented that sequence from the points where the displacement exceeds the upper bound. After preprocessing of the dataset, we add these newly generated sequences as valid into the dataset as well.

All in all, while implementing the system, the maximal diameter of hexagonal cells should be set to the highest spatial displacement between consecutive samples among all sequences of dataset to eliminate issues such as information loss and discontinuity.

# Chapter 6

# Conclusion

The importance of location data privacy has increased as an issue with the extensive usage of location data by mobile devices and applications for location-based services (LBS) and consumer applications. Such service providers generally store the location data in a sequence for each subject in order to optimize their services. However, abusing of this data can lead to identity theft and the disclosure of sensitive, confidential information. As information becomes increasingly decentralized in mobile, cloud based environments, service providers need to focus on the protection of the privacy of this data from adversaries that perform brute-force attacks on their standard encryption schemes. In this work, we propose an system that addresses the problem of storing, retrieving, and aggregating sequential location data in LBS and location-based consumer applications securely.

Although the users are advised to choose high-entropy passwords in order to eliminate brute-force attacks, studies show that brute-force attacks still pose serious risks for individuals and organizations due to the tendency of service users of choosing passwords with low-entropy. Fortunately, we showed that our proposed solution offers a reasonable protection for spatial trajectories against computationally unbounded adversaries regardless of password entropy. When an adversary tries an incorrect key to reveal encrypted spatial trajectory data, our system returns a plausible-looking but incorrect data. In this case, the adversary

cannot eliminate any password from the generated password space, and the brute-force attack will fail.

Our novel solution is based on honey encryption provides secure storage and retrieval of spatial trajectory data. It takes advantage of hexagonal tessellation to calculate probability distribution of location transitions in the study area. The encoding and decoding mechanism works through a 7-ary tree in which we partition the seed space starting from the root to the leaf nodes according to the probability distribution of location transitions. We also come up with techniques that can be useful for location service providers to dynamically aggregate the data. Using these techniques, the DTE structure can be maintained without reconstruction as the dataset changes over time. Moreover, our system provides security against adversaries with side information (point of interests that spatial trajectories possibly pass). We provided implementation of our system and analyze its efficiency with different settings. Eventually, we came to the conclusion that our system is applicable to protect spatial trajectories in LBS and consumer applications. The proposed system can effectively be implemented within predetermined bounds, and service providers that store sequential location data can secure the location privacy of their users without much effort.

# Bibliography


[1] F. C. Commission *et al.*, "Location-based services: An overview of opportunities and other considerations," 2012.

[2] "Location based services market expected to reach $61,897 million by 2022." `http://www.prnewswire.com/news-releases/location-based-services-market-expected-to-reach-61897-million-by-2022-globa html`. Accessed: 2017-09-11.

[3] Y.-A. De Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel, "Unique in the crowd: The privacy bounds of human mobility," *Scientific reports*, vol. 3, p. 1376, 2013.

[4] M. Gruteser and X. Liu, "Protecting privacy, in continuous location-tracking applications," *IEEE Security & Privacy*, vol. 2, no. 2, pp. 28–34, 2004.

[5] S. Lorkowski, P. Mieth, K.-U. Thiessenhusen, D. Chauhan, B. Passfeld, and R.-P. Schäfer, "Towards area-wide traffic monitoring-applications derived from probe vehicle data probe vehicle data," in *Applications of Advanced Technologies in Transportation Engineering (2004)*, pp. 389–394, 2004.

[6] D. J. Patterson, L. Liao, K. Gajos, M. Collier, N. Livic, K. Olson, S. Wang, D. Fox, and H. Kautz, "Opportunity knocks: A system to provide cognitive assistance with transportation services," in *International Conference on Ubiquitous Computing*, pp. 433–450, Springer, 2004.

[7] J. Letchner, J. Krumm, and E. Horvitz, "Trip router with individualized preferences (trip): Incorporating personalization into route planning," in

*Proceedings of the National Conference on Artificial Intelligence*, vol. 21, p. 1795, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.

[8] J. Krumm and E. Horvitz, "Predestination: Inferring destinations from partial trajectories," *UbiComp 2006: Ubiquitous Computing*, pp. 243–260, 2006.

[9] R. Hariharan, J. Krumm, and E. Horvitz, "Web-enhanced gps," in *International Symposium on Location-and Context-Awareness*, pp. 95–104, Springer, 2005.

[10] Z. Huang, E. Ayday, J. Fellay, J.-P. Hubaux, and A. Juels, "Genoguard: Protecting genomic data against brute-force attacks," in *Security and Privacy (SP), 2015 IEEE Symposium on*, pp. 447–462, IEEE, 2015.

[11] D. Florencio and C. Herley, "A large-scale study of web password habits," in *Proceedings of the 16th international conference on World Wide Web*, pp. 657–666, ACM, 2007.

[12] M. Duckham and L. Kulik, "Location privacy and location-aware computing," *Dynamic & mobile GIS: investigating change in space and time*, vol. 3, pp. 35–51, 2006.

[13] B. Schilit, J. Hong, and M. Gruteser, "Wireless location privacy protection," *Computer*, vol. 36, no. 12, pp. 135–137, 2003.

[14] C.-Y. Chow and M. F. Mokbel, "Trajectory privacy in location-based services and data publication," *ACM Sigkdd Explorations Newsletter*, vol. 13, no. 1, pp. 19–29, 2011.

[15] J. Yan, A. Blackwell, R. Anderson, and A. Grant, "Password memorability and security: Empirical results," *IEEE Security & privacy*, vol. 2, no. 5, pp. 25–31, 2004.

[16] A. Juels and T. Ristenpart, "Honey encryption: Security beyond the brute-force bound," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 293–310, Springer, 2014.

[17] B. Kaliski, "Pkcs# 5: Password-based cryptography specification version 2.0," 2000.

[18] "Github weak passwords brute forced." `https://github.com/blog/1698-weak-passwords-brute-forced`. Accessed: 2017-08-11.

[19] S. Gambs, M.-O. Killijian, and M. N. del Prado Cortez, "Show me how you move and i will tell you who you are," in *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS*, pp. 34–41, ACM, 2010.

[20] S. Hansell, "Aol removes search data on vast group of web users," *New York Times*, vol. 8, p. C4, 2006.

[21] A. Narayanan and V. Shmatikov, "De-anonymizing social networks," in *Security and Privacy, 2009 30th IEEE Symposium on*, pp. 173–187, IEEE, 2009.

[22] J. Krumm, "A survey of computational location privacy," *Personal and Ubiquitous Computing*, vol. 13, no. 6, pp. 391–399, 2009.

[23] Y. Zheng and X. Zhou, *Computing with spatial trajectories*. Springer Science & Business Media, 2011.

[24] M. Terrovitis and N. Mamoulis, "Privacy preservation in the publication of trajectories," in *Mobile Data Management, 2008. MDM'08. 9th International Conference on*, pp. 65–72, IEEE, 2008.

[25] A. Hasan, Q. Qu, C. Li, L. Chen, and Q. Jiang, "An effective privacy architecture to preserve user trajectories in reward-based lbs applications," *ISPRS International Journal of Geo-Information*, vol. 7, no. 2, p. 53, 2018.

[26] L. Spitzner, "Honeytokens: The other honeypot," 2003.

[27] L. Spitzner, *Honeypots: tracking hackers*, vol. 1. Addison-Wesley Reading, 2003.

[28] A. Juels and R. L. Rivest, "Honeywords: Making password-cracking detectable," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pp. 145–160, ACM, 2013.

[29] N. Tyagi, J. Wang, K. Wen, and D. Zuo, "Honey encryption applications," *Network Security*, 2015.

[30] J.-I. Kim and J. W. Yoon, "Honey chatting: A novel instant messaging system robust to eavesdropping over communication," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pp. 2184–2188, IEEE, 2016.

[31] J. W. Yoon, H. Kim, H.-J. Jo, H. Lee, and K. Lee, "Visual honey encryption: Application to steganography," in *Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security*, pp. 65–74, ACM, 2015.

[32] C. Pontikakos, T. Glezakos, and T. Tsiligiridis, "Location-based services: architecture overview," in *Proceedings of the International Congress on Information Technology in Agriculture, Food and Environment (ITAFE'05)*, 2005.

[33] C.-Y. Chow and M. F. Mokbel, "Privacy in location-based services: a system architecture perspective," *Sigspatial Special*, vol. 1, no. 2, pp. 23–27, 2009.

[34] A. Juels and T. Ristenpart, "Honey encryption: Encryption beyond the brute-force barrier," *IEEE Security & Privacy*, vol. 12, no. 4, pp. 59–62, 2014.

[35] D. Wang, H. Cheng, P. Wang, X. Huang, and G. Jian, "Zipf's law in passwords," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2776–2791, 2017.

[36] D. M. Powers, "Applications and explanations of zipf's law," in *Proceedings of the joint conferences on new methods in language processing and computational natural language learning*, pp. 151–160, Association for Computational Linguistics, 1998.

[37] "CRAWDAD epfl mobility dataset." https://crawdad.org/epfl/mobility/20090224/. Accessed: 2018-02-16.

[38] J. Freudiger, R. Shokri, and J.-P. Hubaux, "Evaluating the privacy risk of location-based services," in *International conference on financial cryptography and data security*, pp. 31–46, Springer, 2011.