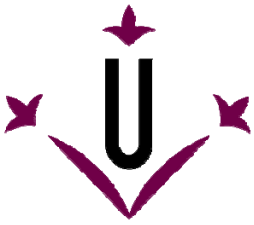


A Privacy-Preserving Pay-by-Phone Parking System

Ricard Garra, Santi Martínez, and Francesc Sebé

***Abstract*—Most cities around the world require drivers to pay for the time they occupy a parking spot. In this way, drivers are encouraged to shorten parking time so that other drivers are given a reasonable chance of finding parking. The traditional way, based on moving to a pay station and placing the issued parking ticket on the dashboard of the car, presents several drawbacks like having to predict in advance the duration of parking or the need to move to the car in case the parking time has to be extended. Over the last few years, several applications permitting to pay through the mobile phone have appeared. Such applications manage detailed information about parking operations so that accurate profiles of parking habits of car owners can be created. In this paper we propose a system to pay for parking by phone which preserves the privacy of drivers in the sense that the information managed by the system is proven not to help an attacker with full access to it to do better that she would do by patrolling the city for collecting information about parked cars.**

***Index Terms*—Cryptography, Pay-by-phone parking, Privacy.**

I. Introduction

THE amount of vehicles in cities is growing every day while it is hardly impossible to increase the amount of on-street parking bays. Restricting the maximum time a vehicle can occupy a parking spot is required to encourage a regular turnover of parking bays and give drivers a reasonable chance of finding parking. An accurate monitoring can only be carried out by installing in-ground sensors that send a notification to a parking officer when a car exceeds the parking time limit. In-ground sensors have been installed in several cities like Melbourne, Westminster or San Francisco. These systems are expensive to install and maintain. In San Francisco, maintenance of a single parking space is beyond $20 per month [5].

A cheaper solution is implemented by requiring drivers to pay for the time they occupy a parking bay. After parking her car, a driver moves to the closest pay station and makes a payment. Some parking machines provide credit card facilities as an additional option to coins. After that, the machine issues a parking ticket that has to be placed on the dashboard of the car. Parking enforcement officers patrol parking zones and monitor for violations which will be punished. Time restrictions are included by limiting the parking duration in a parking ticket. This way of limiting parking time is not accurate since a ticket which is about to expire can simply be replaced with a new one. Nevertheless, paying for parking time encourages most drivers to move their cars as soon as possible.

These systems present several drawbacks:

- Drivers must ensure to have sufficient coins prior to parking (if credit cards are not supported).
- Drivers have to predict (and pay for) the duration of parking in advance. If parking takes less time than predicted, the money corresponding to unused time is lost. If parking time has to be extended, the driver is required to move to the car.
- Moving to the pay station and coming back to the car to place the parking ticket takes time.

Many towns and cities provide the possibility to pay for parking by phone [8], [12]–[16], [19], [21]. A driver installs an app in her mobile phone and creates an account in which she indicates a source for funding such as a credit card number. Upon parking, the driver logs in her account, indicates her car license plate number, the area of the city she has parked in, and the expected duration. After that, a payment for the corresponding amount is performed. Some of these applications permit to interrupt a parking session so that the money corresponding to unused time is refunded. Also, a driver can extend parking time without the need to move to her car.

Parking officers are provided with a mobile device where they can type a car license plate number and check whether a payment for that car is in effect. In such a system, a system server that collects information of all the parking operations is required so that parking officers can query it. Data provided to pay-for-parking applications give rise to privacy concerns since all the parking operations performed by the same car can be linked through the car license plate number. Hence, the information collected by these applications permits to infer the parking habits of car owners.

A. Privacy in car technology

The European Union directive 2010/40/EU (7 July 2010) defines Intelligent Transportation Systems (ITS) as *advanced applications which, without embodying intelligence as such, aim to provide innovative services relating to different modes of transport and traffic management and enable various users to be better informed and make safer, more coordinated, and 'smarter' use of transport networks.*

The inclusion of intelligent devices and radio interfaces on vehicles opens the door to automatic data collection for tracking and monitoring of drivers' behaviour. Security and privacy has been widely addressed in the design of vehicular technology solutions by making use of advanced cryptography.

Privacy-preserving solutions have been proposed for Vehicular Ad-Hoc Networks (VANETs). In [7] the authors present a privacy-preserving system for vehicle-generated announcements based on the use of threshold digital signatures which is secure against external and internal attackers. The proposal [18] employs identity-based group signatures to divide

The authors are with the Department of Mathematics, Universitat de Lleida, C. Jaume II, 69, E-25001 Lleida, Spain (e-mail: garra@matematica.udl.cat; santi@matematica.udl.cat; fsebe@matematica.udl.cat).

a large-scale VANET into easy-to-manage groups and establish liability while preserving privacy. In [6] anonymous credentials are used to protect the privacy of the drivers in a navigation scheme that utilizes the on-line road information collected by a vehicular ad hoc network to guide the drivers to desired destinations in a real-time manner. The authors in [10] propose an authentication framework which makes use of pseudonyms for privacy preservation in which legitimate third parties achieve non-repudiation of vehicles in certain situations like investigations for accidents or liabilities.

Privacy is also an issue in parking space management systems. The system described in [23] gathers information from sensors in parking spaces which is transmitted to drivers' mobile devices so that empty spots are viewed and can be reserved. The application considers privacy by encrypting the wireless communications. Nevertheless, since a parking reservation includes the Electronic License Plate (ELP), system servers are aware of the exact time a car checks in and leaves the parking lot. A similar proposal is presented in [11]. Bilinear pairing cryptography is employed for securing wireless communications. Transactions are performed by an on-board unit (OBU) which is assigned a pseudo-identifier employed to authenticate itself against parking lot road side units (RSU). Since the same pseudo-identifier is employed in all the transactions, profiles can be created.

Transportation systems in which vehicles collect data for services are vulnerable to fake data injection attacks. These attacks are partially avoided by preventing vehicles from sending data about places where they have not been. The authors in [24] present a system in which vehicles construct location proofs from the information received from roadside units. The system provides privacy by not including information about user's identifiers in location proofs.

B. Contribution and plan of this paper

In this paper we present a privacy-preserving pay-by-phone parking system. Privacy is provided by implementing an anonymous e-coin based payment system in which payments are performed for short time intervals. A spent e-coin remains anonymous unless a parking officer located close to a vehicle checks its parking status. In such a case, the spent e-coin can be linked to the car to prove that its driver has actually paid for parking her car. As a result of this query, the available information allows to determine that a payment for the present time has been performed, but it does not permit to get the start and end times of the parking operation. The system also permits a driver who has been fined unfairly to provide cryptographic evidences that a payment had really been made. Last but not least, our system does not require the driver to predict the duration of a parking operation in advance. The driver simply indicates the start and the end times upon parking and removing her car, respectively.

Section I introduces the paper by providing an overview about regulated parking zones together with a review of some papers that describe solutions providing privacy in car technology. After that, Section II surveys current pay-by-phone parking systems. Next, Section III describes the cryptographic tools used by our proposal while Section IV introduces the system and adversary models together with the privacy requirements to be provided by a privacy-preserving pay-by-phone parking system. The novel proposal is detailed in Section V. Its privacy and security properties are analyzed in Section VI. Section VII shows the performance of an Android implementation run over different mobile phones while Section VIII discusses some implementation and deployment challenges. Finally, Section IX concludes the paper.

II. Related work

Nowadays there exist several pay-by-phone parking systems in use. Depending on the method used to specify the duration of a parking period, they may be classified as *start-stop* or *start-duration* systems.

Table I summarizes some of these systems currently in use, the platforms for which they are available, and the method they provide for specifying the duration of a parking period.

TABLE I
Pay-by-phone parking systems

Parking system	Platforms					Parking period
	And.	iOS	Win.	BB	web	
EYSAMobile	✓	✓		✓	✓	start-duration
Pango	✓	✓	✓		✓	start-stop
Parkmobile	✓	✓	✓	✓	✓	start-duration
PayByPhone	✓	✓		✓	✓	start-duration
PayStay	✓	✓			✓	start-duration
Telpark	✓	✓			✓	start-duration

EYSAMobile [8] is a start-duration service with the ability to lengthen or shorten the parking time. It warns the driver when the parking time is about to expire and supports both PayPal and credit card payments. It is implemented as Android, iPhone, BlackBerry and web apps.

Pango Mobile Parking [12] is a start-stop service. It is able to help finding parking. In compatible gated lots and garages the smartphone works as a remote control. Opening the gate to enter activates payment while opening the gate to exit ends the parking session. It is provided as Android, iPhone, Windows and web apps (also a phone number).

Parkmobile [13] is a start-duration service that warns fifteen minutes before time expiration. It supports PayPal, credit card and Parkmobile wallet payments. It is implemented as Android, iPhone, Windows, BlackBerry and web apps.

PayByPhone [15] is a start-duration service that sends reminder messages when the parking time is about to expire. You can extend (but not shorten) a parking session remotely. It allows to pay in compatible parkings and tolls. It is provided as Android, iPhone, BlackBerry and web apps (also a phone number).

PayStay [16] is a start-duration service that warns when the parking period is about to expire. It is implemented as Android, iPhone and web apps (also a phone number).

Telpark [21] is a start-duration service that sends a reminder when the time is about to expire. It permits to extend the parking session remotely. It is provided as Android, iPhone and web apps.

None of the aforementioned pay-by-phone parking systems provides privacy. The system is aware of all the parking operations carried out by drivers so that detailed reports of parking habits can be generated.

To the best of our knowledge, the proposal described in [17] is the only pay-by-phone parking system addressing privacy issues. Like ours, that proposal requires an RFID tag to be placed on cars and payments are made using anonymous e-coins. In [17], when a driver parks her car in a regulated zone, a random identifier shared between the RFID tag and the mobile phone is generated. That identifier is transmitted by the mobile phone to the system server together with an anonymous payment for the required parking time. A parking officer which checks the status of a parked car will read the on-board RFID tag so as to get its current identifier. Then he will query the server to check whether a valid payment linked to that identifier has been received. Since the parking officer is close to the car, she can see its license plate number. At this moment the system server and the parking officer have enough information to link a car license plate number to the exact begin and end times of a parking operation. Hence a parking operation only remains private if the parked car is not checked by a parking officer. In our proposal, the only information obtained from a parking status query is a boolean indicating whether a payment for the current time has been performed (the start and end times stay unknown). The system in [17] permits to extend the parking time remotely from the mobile phone but it does not allow to recover the money corresponding to non-used parking time in case the parking operation takes less time than expected. In our solution, the driver pays exactly for her parking duration. Another feature offered by our system and not provided by [17] is the possibility to provide cryptographic evidence that a payment was performed in case a driver is fined unfairly.

III. Preliminaries

This section describes the cryptographic methods required by the proposed system.

A. Hash-based message authentication codes

A hash-based message authentication code [3] (HMAC) is a construction for calculating an authentication code of a message M given a secret key K. The resulting code will be denoted $\mathrm{HMAC}_K(M)$. An HMAC is a keyed cryptographic one-way hash function. It offers the one-way and collusion-secure properties of hash functions with the additional property that the HMAC digest of a message M can only be computed if the secret key K is known.

When a message M is sent together with $\mathrm{HMAC}_K(M)$, a party who knows the secret key K, upon receiving M, can compute the authentication code by itself and check that the result equals the accompanying code. A proper HMAC validation provides *data integrity* in the sense that the message can not have been modified during its transmission and *authentication* since the accompanying code can only have been computed by a party who knows the secret key.

For an HMAC to be secure it is required that, given M and $\mathrm{HMAC}_K(M)$, it is infeasible to find K. Also, given $\mathrm{HMAC}_K(M)$ and K, it is infeasible to get M. Both the secret key and the generated authentication code should be at least 128 bits long.

An HMAC is implemented as a procedure involving the computation of two cryptographic hash digests such as SHA-1 or SHA-256.

B. RSA digital signatures

RSA [20] is a widely known public key cryptosystem whose security holds on the assumed intractability of the integer factorization problem. Our proposal employs it for digital signature computation. An RSA digital signature scheme is composed of three procedures: *private/public key pair generation*, *signature computation* and *signature validation*.

A digitally signed message provides *authentication* (the message was created by somebody who knows the secret key related to the public key under which the signature is validated), *non-repudiation* (the signer can not deny having signed the message) and *integrity* (the message has not been altered after being signed).

Throughout the paper, the private and public keys of an entity E will be denoted KPriv_E and KPub_E, respectively. A digital signature on a message M signed under the key pair of E will be denoted $\mathrm{Sign}_E(M)$.

When an RSA public key has a small public exponent (usually, 65537), the resulting signatures can be verified fast.

C. RSA blind signatures

A blind signature [4] is a form of digital signature in which the signer and the message owner are different parties. After an execution of a blind digital signature protocol, the message owner obtains a signer's digital signature on her message while the signer gets no information about the message she actually signed. The RSA cryptosystem provides a simple method for computing blind signatures.

Key generation in RSA is a time-consuming task due to the need to generate two large prime numbers. Hence, the RSA cryptosystem is not a good option in systems requiring the generation of a large amount of public keys. Elliptic curve cryptography overcomes the mentioned drawback of RSA.

D. Cryptography over elliptic curves

An ElGamal-like public key cryptosystem can be built over a prime order subgroup of the group of points of an elliptic curve [9]. Such a cryptosystem requires a *setup* procedure in which a proper elliptic curve is chosen. After the setup has been carried out, a private/public key pair can be generated very fast. The Elliptic Curve Digital Signature Algorithm (ECDSA) [1] is a standard for computing digital signatures on an elliptic curve cryptosystem.

E. Trusted timestamping

A trusted timestamp [2] is a timestamp issued by a trusted party acting as a timestamp authority (TSA). A trusted timestamp is used to prove the existence of a certain piece of data before a certain point in time.

A timestamp on a message M is generated by first computing a hash digest of M, $\mathcal{H}(M)$. That digest is transmitted to the timestamp authority which will concatenate a timestamp to that hash and then compute the hash of this concatenation. The obtained digest is digitally signed with the private key of the TSA. The resulting signature is sent back to the timestamp requester.

A timestamp on M is checked by verifying that the hash of the concatenation of $\mathcal{H}(M)$ and the timestamp were actually signed by the TSA.

F. Anonymous communications on the Internet

IP datagrams transmitted through the Internet include the IP address of the source device. That information is required for addressing the response datagrams. In this way, datagrams sent from the same device, although belonging to different TCP connections, can be linked through the IP source address field.

Tor [22] is a software for enabling anonymous communications. Tor directs Internet traffic through a free, worldwide, volunteer network to conceal a user's location and usage from anyone conducting network surveillance or traffic analysis. The destination server receives the data from a randomly selected Tor relay so that the real source device remains unknown to it. Data transmitted in this way is said to be transmitted through an *anonymous channel*.

Tor is available for mobile devices. The Orbot package implements a Tor client for mobile devices running Android.

IV. System and adversary models

This section first presents the system and adversary models. After that, the privacy requirements to be satisfied by a privacy-preserving pay-by-phone parking system are stated.

A. System model

Pay-by-phone parking systems implement a system model composed of four actor types:

- *Mobile application:* It is installed and runs in the mobile device of drivers. Drivers use it to manage the credit in their accounts (in prepaid systems) and to pay for parking operations.
- *System server:* This is an on-line platform accessed by drivers, via the mobile application, to purchase credit or to pay for a parking operation. It manages the information about parking operations. This platform may include several machines performing different tasks (our proposal includes a timestamp server).
- *Parking officer:* He patrols the city carrying a mobile device through which he queries the system server to check the parking status of cars.
- *Car:* Each car has an identifier used by parking officers to check its parking status. Most pay-by-phone parking systems use the license plate number as identifier. In [17] and in our proposal, the identifier is a pseudo-random binary sequence transmitted via RFID.

Figure 1 depicts the main interactions among these actors. A mobile application, upon indication of its owner, contacts the system server to purchase credit or to pay for parking. In systems providing privacy about drivers' parking habits, the communications carried out during a parking operation payment have to employ an anonymous channel. A parking officer checks the parking status of a car by first getting its identifier and then querying the system server to check whether a valid payment is in effect.

Other interactions are possible. In our proposal the mobile application can also contact the system server to complain for an unfairly received fine.

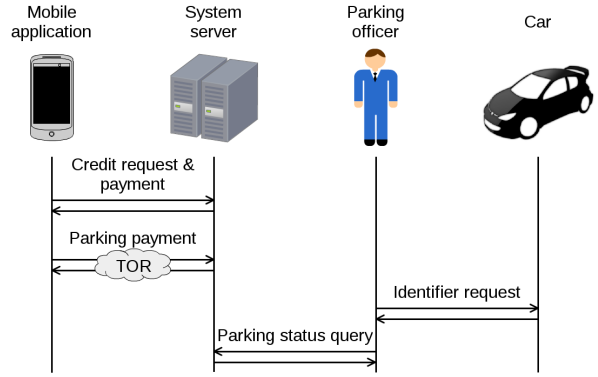


Fig. 1. System model.

B. Adversary model

As a fundamental aspect, we assume the employed cryptography provides computational security in the sense that no party has enough computing power to perform brute force attacks against the cryptographic primitives.

Regarding an adversary trying to compromise drivers' privacy, the following assumptions are made:

- An adversary cannot corrupt the application running on the mobile phone nor an eventual on-board device placed in cars. The mobile app and on-board devices act as specified by the system protocols and do not leak any private information about the data they store nor about the internal computations they perform.
- An adversary has full access to all the information received and managed by the system server and parking officers, but it cannot corrupt their behaviour. In this sense, the system server and parking officers are honest but curious entities. They do not deviate from the protocol but can collaborate with an adversary who is trying to get information about drivers' parking habits.

Drivers are untrusted entities who might try to get parking time without paying for it. To that end, several drivers could collude and/or transmit misgenerated data. In fact, the system server can not be sure that an entity communicating with it is running an original app.

C. Design objectives

In our opinion, a privacy-preserving pay-by-phone parking system should provide the following privacy features:

1) A payment performed for parking a vehicle remains anonymous unless a parking officer located close to the car checks its payment status or the car owner complains for an unfairly received fine.
2) The only information obtained by a parking officer with full access to all the information managed by the system regarding a parked car is a boolean indicating whether a payment valid for the present time has been properly performed.
3) A driver can prove to a third party that she performed a payment for a period including a given time instant. In that case, no information about other time instants is revealed.

The system must allow to determine whether a given car has paid for parking or not. Requirement 1 states that checking can only be performed by a parking officer who is close to the vehicle. In this way, the system does not provide any help to automatic creation of parking profiles. The information collected by the system does not help an attacker with full access to it to do better that she would do by patrolling the city for collecting information about parked cars. Requiring a parking officer to be near the vehicle implies that the parking status of a car can not be checked by just typing the license plate number in an application. Some kind of dynamic pseudonym only obtainable when you are close to the vehicle is required for checking a parking status.

Note that Requirement 2 causes that a parking officer can not determine when a car was parked neither how long the parking operation will take.

Requirement 3 is for allowing users to prove a payment had been performed in case they were fined unfairly. In such a case, no information about other time instants is leaked.

V. Our proposal

A. System overview

In our proposal a driver has to install an app in her mobile phone and place an on-board device in her car. Both devices share a secret key.

The app manages an electronic wallet which contains electronic coins used for paying for parking time. When the wallet is about to run out of e-coins, the user may contact the system server and request a bunch of new electronic coins (the app could be configured to perform this request automatically). These coins are paid by credit card or using some on-line payment system such as PayPal.

Upon parking the car in a regulated zone, the driver logs in her app and indicates the beginning of a parking operation. At this moment, the app contacts the system server and performs an e-coin payment operation. Each e-coin payment is for a short time interval (e.g., 10 minutes). The app regularly contacts the server (when the previous payment is about to expire) and pays for the next time slot. A time slot is represented as an integer (for instance, amount of 10 minutes intervals since the beginning of year 2000). When the system server receives an e-coin, it requests a time authority to timestamp it and sends the timestamp back to the app which will store it. When the driver removes the car from the parking bay, she indicates the app to stop performing regular payments.

A description of the system components is next provided:

- *Mobile application:* This is an app installed in the mobile device of drivers. Its functionalities are:
  - Request and pay for a bunch of new e-coins when the wallet is about to run out of e-coins.
  - Start making periodic payments when a parking operation begins and stop doing them when it concludes.
  - Permit the driver to complain about an unfairly received fine.

  All these operations are made against the system server. E-coin payments are transmitted through an anonymous channel.
- *On-board device:* This is a device located in the car which is readable via RFID by the mobile device carried by parking officers. Upon a parking officer request, it responds by transmitting its *current time pseudonym*. This device incorporates an internal clock and can communicate via NFC with the driver's mobile phone for setup.
- *Parking officer:* A parking officer carries a device with an RFID reader which can query the on-board device of a car and ask for its current pseudonym. After getting the pseudonym the mobile device contacts the system server to check whether a valid payment has been performed for the checked car.
- *Timestamp server:* It timestamps the payments performed by the mobile app of drivers so that, if necessary, drivers can prove that a certain payment was made at a given time. The data to be timestamped is received from the system server.
- *System server:* It provides anonymous e-coins to mobile apps. It also receives e-coin payments from the mobile apps (through an anonymous channel), requests the timestamp server to timestamp them and sends each timestamp back to the corresponding mobile app. Parking officers query it for checking the payment status of parked cars. These queries are made using the current time pseudonym of cars.

Note that our proposal matches the system model described in Section IV-A

B. System description

Our system is composed of the following procedures:

- System parameters establishment
- Setup
- E-coins request
- Time slot payment
- Parking status query
- Fine complaint

1) System parameters establishment: This procedure is carried out when the company providing the pay-for-parking service starts to operate. That company manages a system

server which, prior to operating, has to set some cryptographic parameters:

- Generate an RSA private/public key pair, $\mathrm{KPriv}_{Server}/\mathrm{KPub}_{Server}$. The server public key must be certified by some certificate authority.
- Set an elliptic curve (together with a generator of its large primer order subgroup and all the required parameters) suitable for cryptographic uses.

The system includes a timestamp authority which manages a server that is going to timestamp the spent e-coins. This authority possesses an RSA private key KPriv_{TSA} and the corresponding public key KPub_{TSA} with an appropiate digital certificate. This key pair is employed for issuing timestamps.

2) Setup: This procedure is run when a driver wishes to start using the system. The driver has to download and install the application in her mobile device and acquire an on-board device. On-board devices are manufactured so that each one comes with a stored secret key K_{NFC} which is communicated to the driver in a sealed envelope. In a first step, the driver types the license plate number Lic, the key K_{NFC}, and the data required for purchasing e-coins (such as a credit card number) into the mobile application so that they are stored there.

Next a secret key s, shared between the on-board device and the mobile app is established. We propose a method in which the driver chooses to run a "Parameter setting" option in the mobile application. The application generates and stores a random secret key s and then composes a "Parameter setting message" which includes the secret s, the car license plate number Lic, the current time $Time$, and a secure authentication code $\mathrm{HMAC}_{K_{NFC}}(s||Lic||Time)$ only computable if you are in possession of K_{NFC} (see Section III-A). That setting message is then encrypted in AES under key K_{NFC} and transmitted to the on-board device via NFC. Upon receiving that message, the device decrypts it and checks the authentication code. If the verification is successful it stores the received s and Lic. It also sets its internal clock to the received $Time$.

The system should include a method allowing the driver to modify the K_{NFC} key.

3) E-coins request: When the mobile application is run for the first time or it is about to run out of e-coins, the user can contact the system server and ask for a bunch of new e-coins. Then a payment by credit card or some other payment method will be performed. The value (and price) of each e-coin corresponds to the price of parking during a time slot (e.g., 10 minutes) in a specific zone.

An e-coin is generated as follows:

1) The mobile app, using the elliptic curve set by the system server during the parameter establishment phase, randomly generates an elliptic curve secret key KPriv_{coin} and its corresponding public key KPub_{coin} (see Section III-D).
2) Next, the mobile app and the system server engage in an RSA blind signature protocol (see Section III-C) so that, as a result, the mobile app obtains a system server RSA signature on its elliptic curve public key. An e-coin is a tuple
$$\{\mathrm{Sign}_{Server}(\mathrm{KPub}_{coin}), \mathrm{KPub}_{coin}, \mathrm{KPriv}_{coin}\}.$$
3) The app stores the generated e-coin.

The previous e-coin withdrawal protocol is executed as many times as the amount of requested e-coins.

4) Time slot payment: When the driver parks her car in a regulated zone, she logs in the mobile app and indicates the beginning of a parking period. From now on, the application will spend an e-coin at the beginning of each time slot.

An e-coin is spent as follows:

1) From the current time, the application generates an integer indicating the current time slot, $Slot$ (amount of 10 minute intervals since year 2000).
2) The app generates a *current time intermediate identifier* by computing $h_{Slot} = \mathrm{HMAC}_s(Slot)$. The *current time identifier* is then generated as $ID_{Slot} = \mathrm{HMAC}_{h_{Slot}}(Slot||Lic)$.
3) Then, it takes an unspent e-coin, $\{\mathrm{Sign}_{Server}(\mathrm{KPub}_{coin}), \mathrm{KPub}_{coin}, \mathrm{KPriv}_{coin}\}$, from the phone wallet, and using secret key KPriv_{coin}, computes the elliptic curve digital signature $\mathrm{Sign}_{coin}(ID_{Slot})$.
4) After that, it sends $\mathrm{Sign}_{Server}(\mathrm{KPub}_{coin})$, KPub_{coin}, ID_{Slot} and $\mathrm{Sign}_{coin}(ID_{Slot})$ to the system server through an anonymous channel.
5) The system server:
   a) Checks that $\mathrm{Sign}_{Server}(\mathrm{KPub}_{coin})$ is a valid signature on KPub_{coin} verifiable under its RSA public key KPub_{Server}.
   b) Checks that it has not accepted a previous payment with an e-coin containing KPub_{coin}.
   c) Checks that $\mathrm{Sign}_{coin}(ID_{Slot})$ is a valid signature on ID_{Slot} verifiable under the elliptic curve public key KPub_{coin}.
6) If all the previous checkings are satisfied, it requests the TSA to timestamp ID_{Slot}. The received timestamp is forwarded to the mobile app which receives it through the anonymous channel and stores it.
7) Finally the system server stores a tuple with KPub_{coin}, ID_{Slot}, $\mathrm{Sign}_{coin}(ID_{Slot})$ and $\mathrm{TimeStamp}_{TSA}(ID_{Slot})$ in a database containing all the spent coins.

This payment process will be automatically repeated every time interval until the driver indicates the app to stop.

5) Parking status query: Parking officers patrol parking areas and check that a payment has been performed for the parked cars.

This is done as follows:

1) The parking officer types the car license plate number, Lic, into her device. Then she approaches it to the car and queries its on-board device by performing an RFID query.
2) Upon receiving that query, the device reads its internal clock and computes the current time slot, $Slot$. From $Slot$, the secret key s, and the license plate number Lic, the on-board device computes h_{Slot} and ID_{Slot} as

described in Step 2 of the time slot payment procedure (Section V-B4).
3) The on-board device responds to the query by returning $Slot$, h_{Slot} and ID_{Slot}.
4) Upon receiving the response, the mobile device:
   a) Checks that the received $Slot$ corresponds to the current time.
   b) Checks that the received ID_{Slot} equals $\mathrm{HMAC}_{h_{Slot}}(Slot||Lic)$.
   c) Queries the system server to verify that an e-coin linked to ID_{Slot} has been spent. If the response is negative, the car will be fined.

6) Fine complaint: If a driver is fined during a time slot for which she had actually made a payment, she can prove that she had actually paid by means of the following procedure carried out through the mobile app.

1) Introduce the date and time of the fine so that the app can compute the time slot, $Slot$, the fine was made.
2) Compute the intermediate identifier as $h_{Slot} = \mathrm{HMAC}_s(Slot)$.
3) Send $Slot$, h_{Slot}, Lic, and the timestamped ID_{Slot} to the system server.
4) From the received data, the system server computes $ID_{Slot} = \mathrm{HMAC}_{h_{Slot}}(Slot||Lic)$ and verifies the timestamp on ID_{Slot}. If the previous checking is satisfied, the fine is removed.

VI. Security analysis

In this section, the security of the presented proposal is analyzed. The attacker model assumed in Section IV-B states that the on-board device and the mobile app can not be corrupted by an adversary. In our proposal, this assumption implies that:

- The on-board device:
  - It only transmits h_{Slot} and ID_{Slot} for the current time slot, $Slot$, upon reception of a query from a close enough parking officer RFID reader (its internal clock can not be manipulated).
  - It does not leak any information about the stored secret key s nor about the internal computations it performs.
- The application running on the mobile phone:
  - It properly runs all the system procedures as described in the paper and upon indication of the driver.
  - It does not leak any information about the stored data nor about the internal computations it performs but that transmitted as specified by the system protocols.
  - When required, it can communicate anonymously with the system server (see Section III-F).

Trust on the on-board device can be achieved by making it incorporate tamper–resistant storage and computation media, like a smart card processor. For the mobile app to be trusted, it must be audited by an external entity to check it properly implements the system procedures. Also, the data stored by the app should not be accessible to eventual malware which has broken into the phone. Security measures incorporated by nowadays mobile phones like process isolation (sandboxing) and encrypted filesystems contribute to make this assumption realistic.

The attacker model also assumes that the system server (including the timestamp server) and the parking officers are honest but curious. Next we discuss the feasibility of this assumption. In our system, the mobile app interaction with the system and timestamp servers consists on the reception of signed data (a signed e-coin during an e-coin request and a timestamped piece of data after a time slot payment) whose validity is easily checked by the mobile app. Hence, these servers cannot misoperate without the app immediately noticing it. Eventual attacks based on interrupting the system procedures execution, or issuance of unfair parking fines by the parking officers would cause the drivers to complain to the city council asking for an alternative service provider. Hence, assuming that in case of corruption they take a honest but curious behaviour is realistic.

A. Privacy analysis

We consider an attack against the privacy of a driver is successful when a malicious coalition composed of the system server, the timestamp authority, and some parking officers is able to determine her car was parked at a given time slot without none of the parking officers being located close to the car.

The following lemmas and theorem state the previous privacy breach is not possible under the assumed adversary model.

Lemma 1: If the intermediate identifier h_{Slot} remains secret, a spent electronic coin can not be linked to a car license plate number.

Proof. During an e-coin request phase, the mobile device of a driver withdraws a set of e-coins. Each e-coin is randomly generated in driver's device so that the server gets no information about it at this time. After that, the e-coin public key KPub_{coin} is blindly signed by the server. A blind signature protocol guarantees that the server gets no information about the signed data (see Section III-C). Hence, no information about the e-coin is obtained as a result of an e-coin withdrawal operation. Since e-coins are generated independently at random, there exists no relation among them. The mobile device is assumed to leak no information about the generated coins.

When an e-coin is spent (see Section V-B4), the e-coin public data is transmitted anonymously to the system server so that the data received by the server can not be related to the mobile device which is transmitting it nor to any coin previously spent by that device. That e-coin is linked to the current time identifier ID_{slot} via a digital signature (step 3). Next, we show that ID_{slot} can not be related to the car.

Given ID_{slot}, the server knows the value of $Slot$ and may be interested in checking whether that identifier corresponds to a given license plate number Lic. Since $ID_{slot} = \mathrm{HMAC}_{h_{Slot}}(Slot||Lic)$, checking for that relation would require a brute force search for h_{Slot} which is infeasible (see Section III-A). Another possibility would be to

link h_{Slot} and $Slot$ via the relation $h_{Slot} = \mathrm{HMAC}_s(Slot)$, but this requires a brute force search for s which is also unfeasible. Both the on-board and the mobile phone devices are assumed not to leak any information about the secret key s nor about h_{Slot}. □

Lemma 2: If the secret key s is kept secret, knowledge of an intermediate identifier h_{Slot} for a given time slot, $Slot$, does not provide any advantage in the computation of $h_{Slot'}$ for another time slot, $Slot'$.

Proof. The value h_{Slot} is computed as $h_{Slot} = \mathrm{HMAC}_s(Slot)$. Since finding s from h_{Slot} and $Slot$ is unfeasible, the computation $h_{Slot'} = \mathrm{HMAC}_s(Slot')$ can not be carried out unless s is known. The secret key s is only stored in the on-board device and in the mobile phone device. Both devices are assumed to keep it secret. □

Theorem 1: The proposed system fulfills the privacy requirements enumerated in Section IV-C.

Proof. Requirement 1 states that a spent e-coin remains anonymous unless a parking officer checks a parking status or a fine complaint operation takes place. An e-coin is spent linked to a time identifier ID_{Slot}. From Lemma 1, we know that ID_{slot} can not be linked to a car license plate number unless some additional information is provided. Next, we explain the two only cases in which such information is made available.

When a car parking officer checks for the status of a car, it queries the on-board device and gets h_{Slot} and ID_{Slot} as response. The parking officer can then determine the car plate number linked to ID_{Slot} because he is in front of the car whose on-board device has transmitted ID_{Slot}. Hence, the parking officer and the system server have enough information to link a spent coin stored in the system server database to a car license plate number. Such relation permits to infer that the car with Lic was parked at a given time. But this information is already known by the parking officer since he is located in front of the parked car. The identifier ID_{Slot} is related to a car only during a given time slot (10 minutes). Hence, the only information obtained is that the car with plate number Lic performed a payment valid for a given time slot, $Slot$. Lemma 2 states that, as long as s remains secret, revealing h_{Slot} does not provide information about other time slots. During the car parking status query, no values $h_{Slot'}$ nor $ID_{Slot'}$ for other time slots are revealed by the on-board device, hence Requirement 2 is fulfilled.

When a driver complains for an unfair fine received during a time slot, $Slot$, the mobile application computes h_{Slot} and sends this value, together with $Slot$ and Lic to the system server. Then the system server computes ID_{Slot} and checks that the app sent a proper timestamp for it which proves that an appropiate payment for ID_{Slot} was performed. In that case, the system server can search its database for a spent coin related to ID_{Slot} which can be linked to Lic. Lemma 2 states that, as long as s remains secret, revealing h_{Slot} does not provide information about other time slots. During a fine complaint, no value $h_{Slot'}$ for other time slots is revealed by the mobile phone, hence Requirement 3 is also fulfilled. □

B. E-coin system security analysis

Theorem 1 proves that the proposed system fulfills its privacy requirements. Next we comment the fulfillment of some additional security properties that have to be provided by any e-coin based payment system: *unforgeability* and *security against double spending*.

Lemma 3: The e-coin system employed by our proposal is unforgeable.

Proof. An e-coin system is unforgeable if valid e-coins cannot be created by dishonest entities. In our system, an e-coin is valid if and only if its public key component (KPub_{coin}) has been properly signed by the server. Hence, an e-coin can only be generated if the server takes part in the process by (blindly) signing its public key component. Obviously, the server will only participate in the creation of e-coins after receiving an appropiate payment. Since an RSA signature over hashed data is non-malleable, availability of server signatures on previous e-coins can not be used to forge new ones. Obviously, the server's private key is assumed to be kept secure. □

Lemma 4: The e-coin system employed by our proposal is secure against double-spending.

Proof. Double-spending is a fraud in which an e-coin is spent more than one time. In our system, the server stores the spent e-coins in a database. If a previously spent e-coin is received again, the server will detect the situation since it will find that the public key component of the newly received e-coin is already stored in its database.

In case of dispute, the server can show that an e-coin was previously spent by showing ID_{slot}, the public key KPub_{coin}, the signature $\mathrm{Sign}_{coin}(ID_{Slot})$, and $\mathrm{TimeStamp}_{TSA}(ID_{Slot})$. The timestamp provides the exact time the e-coin was spent for the first time. Moreover, the signature $\mathrm{Sign}_{coin}(ID_{Slot})$ can only have been generated by the party who created the e-coin since its generation requires knowledge of the secret key KPriv_{coin}. In this way, the possession of $\mathrm{Sign}_{coin}(ID_{Slot})$ by the server indicates that the party who created that e-coin actively participated in its spending process.

Since e-coins are generated by the mobile phone app, it could happen that the apps of two different drivers generate exactly the same e-coin by randomly choosing the same private key (step 1 in Section V-B3). Since the private key is a large randomly chosen integer (at least 224 bits), the first collision is expected to happen after the generation of at least 2^{112} e-coins. Hence, a collision will rarely happen. The low value of an e-coin (the price of 10 minutes of parking time) permits to implement a solution in which drivers accept to remove an e-coin if that e-coin is reported to have been spent previously. Obviously, if that situation is repeated several times, the driver will have to complain. □

VII. Experimental results

The designed system has been implemented as an Android app. We have then measured the time required for e-

coin generation and time slot payment. These are the two procedures that will be run frequently on drivers' mobile phone. The performance of the remaining procedures is not so relevant. For instance, the setup procedure is run just once after installing the application, while fine complaint will rarely be required. In any case, their complexities are similar to those of the measured procedures so that times with similar magnitude would be obtained.

TABLE II
MOBILE APPLICATION RUNNING TIMES (IN MILLISECONDS).

Mobile phone	Request 12 e-coins		Pay 12 time slots	
	Parallel	Serial	Parallel	Serial
HTC EVO 3D	3009	4958	2961	4874
S. Galaxy S III mini	2250	2835	2218	3043
LG Nexus 4	2977	4160	2932	4057
LG Nexus 5	559	1762	470	1428
LG Nexus 5X	326	553	264	539
Huawei Nexus 6P	300	520	218	479

Our implementation employs 2048–bit RSA and 224–bit elliptic curve keys which, as of 2016, are considered secure. We have measured the time required to request 12 e-coins and the time to pay for 12 time slots, both in serial and in a threaded parallel version of each procedure on a collection of Android phones with different capabilities. The following list summarizes their processors and Android versions. The RAM memory sizes are not included since the app requires very little memory.

- HTC EVO 3D: Qualcomm MSM8660 Snapdragon S3 (Dual-core 1.2 GHz Scorpion), with Android 4.0.3.
- Samsung Galaxy S III mini: NovaThor U8420 (1.0 GHz dual-core Cortex-A9), with Android 4.2.2.
- LG Nexus 4: Qualcomm APQ8064 Snapdragon S4 Pro (Quad-core 1.5 GHz Krait), with Android 5.1.1.
- LG Nexus 5: Qualcomm MSM8974 Snapdragon 800 (Quad-core 2.3 GHz Krait 400), with Android 6.0.1.
- LG Nexus 5X: Qualcomm MSM8992 Snapdragon 808 (Quad-core 1.44 GHz Cortex-A53 & dual-core 1.82 GHz Cortex-A57), with Android 6.0.1.
- Huawei Nexus 6P: Qualcomm MSM8994 Snapdragon 810 (Quad-core 1.55 GHz Cortex-A53 & Quad-core 2.0 GHz Cortex-A57), with Android 6.0.1.

Table II shows the measured times for requesting 12 e-coins (serial and parallel) and paying for 12 time slots (also serial and parallel) for each device. It has to be taken into account that the Nexus devices used in our experiments manage an encrypted filesystem which negativelly affects their performances, specially in the Nexus 4 model. Our experiments do not include the time due to delay in network communications since this is an aspect depending exclusively on the communication network. With the adventure of 4G networks, network delay will surely become negligible in the near future.

We have observed that the running time strongly depends on the device computation power. The slowest speed is obtained for the HTC EVO 3D mobile phone which was released on July, 2011. More up-to-date devices, like LG Nexus 5X and Huawei Nexus 6P, both released in 2015, provide ten times faster times. In all the cases, the measured computation times prove that the proposed system is feasible to be implemented for current mobile devices.

Regarding the complexity at the server side, the system has been tested on a computer with an Intel i5-4460 3.2 GHz processor. In our experiments, a single core was able to compute more than 50,000 RSA blind signatures per second, which is the cryptographic operation performed by the server when generating an e-coin. Regarding the reception of e-coin payments, a single core could validate up to 170 e-coins per second. In a quad-core parallel implementation, we achieved over 200,000 RSA signature computations and 680 e-coin validations per second.

VIII. IMPLEMENTATION AND DEPLOYMENT CHALLENGES

The system server should be deployed on a computer with a reliable Internet connection placed in a secure environment protecting it against eventual attacks aiming to disrupt it. Hence, placing it in a data center with intrusion detection and protection mechanisms is recommended. We also recommend to contract the services provided by an online payment provider to deal with the payments received from the drivers when acquiring e-coins. The timestamp server could be deployed on the same computer but it would be better to place it on a separate machine accessible only from the system server.

Regarding the mobile app (run by drivers), it must be run on a mobile phone with an Internet connection. The system makes use of direct client-server communications against the server during the e-coin request and fine complaint procedures, and a client-server anonymous communication during time slot payment procedure. Software implementing anonymous communications for mobile phones is currently available. The mobile device should be NFC enabled, which is a common feature in nowadays mobile phones, for communicating with the on-board device during setup.

Parking officers carry a mobile device with an Internet connection and an HF RFID reader capable of querying the on-board device. Such devices are already available.

The most challenging part is the on-board device to be placed in cars. That device should be composed of a smartcard–type tamper–resistant processor able to compute HMAC digests and AES decryption operations. It also carries an internal clock which requires the device to be fed through its own batteries. Regarding communications, the device has to be able to act as a receiver for NFC communications (run during the setup procedure) and respond to RFID queries coming from parking officers.

As for the cryptographic operations, there already exist smartcard processors implementing the required operations. Some of them support 128-bit AES encryption/decryption and SHA-1 among others. NFC operates at the 13.56 MHz frequency, also used by High-Frequency (HF) RFID tags. Hence, an HF RFID enabled device can communicate both with NFC and HF RFID readers. In this way, the on-board device only needs to carry an HF RFID antenna. The need of an internal clock requires that device to include a battery. We conclude

that the on-board device could be easily manufacturable with current technology.

IX. Conclusion

A privacy-preserving pay-by-phone parking system has been presented. From the driver's point of view, the system is composed of two components: an RFID and NFC enabled on-board device which is placed in the car, and an app which is installed in the mobile phone.

The app manages an electronic wallet which is loaded with e-coins. When the driver parks her car in a regulated area, the mobile app starts making periodic e-coin payments for short time intervals until the car is removed from the parking spot.

The system has been proven to provide privacy by not allowing the creation of profiles about drivers' parking habits. The system is also secure against e-coin forgery and double-spending and permits a driver who has been fined unfairly to prove, by providing cryptographic evidences, that a payment had really been made.

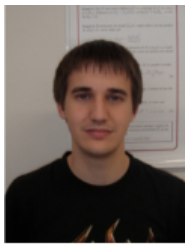
In the future we plan to investigate the design of the app focusing on its usability. Together with the design of the graphical interface, we will also investigate solutions permitting to use the application even when the driver expects to be out of coverage during part of the time her car is parked.

Acknowledgment

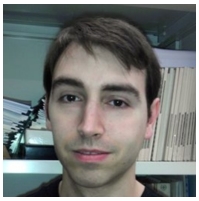
The authors acknowledge partial support by the Spanish Government under project MTM2013–46949–P and by the Government of Catalonia under grants 2014SGR–1666 and 2016FI–B1 00155.

References

[1] Accredited Standards Committee, American National Standard X9.62-2005, Public Key Cryptography for the Financial Services Industry, The Elliptic Curve Digital Signature Algorithm (ECDSA), 2005.
[2] C. Adams, P. Cain, D. Pinkas, and R. Zuccherato, "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)", *Request For Comments – RFC 3161*, 2001.
[3] M. Bellare, R. Canetti, and H. Krawczyk, "Keying hash functions for message authentication", *Proc. of Crypto'96*, Lecture Notes in Computer Science, vol. 1109, Springer-Verlag, 1996, pp. 1–15.
[4] D. Chaum, "Blind signatures for untraceable payments", *Proc. of Crypto'82*, Springer US, 1983, pp. 199–203.
[5] X. Chen, E. Santos-Neto, and M. Ripeanu, "Crowd-based smart parking: a case study for mobile crowdsourcing", *Proc. of MobilWare'2012*, 2013.
[6] T.W. Chim, S.M. Yiu, L.C.K. Hui, and V.O.K. Li, "VSPN: VANET-based secure and privacy-preserving navigation", *IEEE Transactions on Computers*, vol. 63, no. 2, pp. 510–524, 2014.
[7] V. Daza, J. Domingo-Ferrer, F. Sebé, and A. Viejo, "Trustworthy privacy-preserving car-generated announcements in vehicular ad hoc networks", *IEEE Transactions on Vehicular Technology*, vol. 58, no. 4, pp. 1876–1886, 2009.
[8] "EYSAMobile."[Online]. Available: http://www.eysamobile.com
[9] D. Hankerson, A. Menezes, and S.Vanstone, Guide to Elliptic Curve Cryptography, Springer, New York, 2004.
[10] J. Li, H. Lu, and M. Guizani, "ACPN: a novel authentication framework with conditional privacy-preservation and non-repudiation for VANETs", *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 938–948, 2015.
[11] R. Lu, X. Lin, H. Zhu, and X. Shen, "An intelligent secure and privacy-preserving parking scheme through vehicular communications", *IEEE Transactions on Vehicular Technology*, vol. 59, no. 6, pp. 2772–2784, 2010.
[12] "Pango Mobile Parking."[Online]. Available: http://www.mypango.com/
[13] "Parkmobile."[Online]. Available: http://www.parkmobile.com
[14] "ParkRight."[Online]. Available: https://www.westminster.gov.uk/parkright
[15] "PayByPhone."[Online]. Available: https://www.paybyphone.com
[16] "PayStay."[Online]. Available: https://paystay.com.au/
[17] P.A. Pérez-Martínez, A. Martínez-Ballesté, and A. Solanas, "Privacy in smart cities - a case study of smart public parking", *Proc. of Intl. Conf. on Pervasive and Embedded Computing and Communication Systems*, 2013, pp. 55–59.
[18] B. Qin, Q. Wu, J. Domingo-Ferrer, and L. Zhang, "Preserving security and privacy in large-scale VANETs", *Proc. of ICICS 2011*, Lecture Notes in Computer Science, vol. 7043, Springer-Verlag, 2011, pp. 121-135.
[19] "RingGo."[Online]. Available: https://www.myringgo.co.uk
[20] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.
[21] "Telpark."[Online]. Available: https://www.telpark.com
[22] "Tor Project: Anonymity Online." [Online]. Available: https://www.torproject.org
[23] G. Yan, W. Yang, D.B. Rawat, and S. Olariu, "SmartParking: a secure and intelligent parking system", *IEEE Intelligent Transportation Systems Magazine*, vol. 3, no. 1, pp.18–30, 2011.
[24] Y. Zhang, C.C. Tan, F. Xu, H. Han, and Q. Li, "VProof: lightweight privacy-preserving vehicle location proofs", *IEEE Transactions on Vehicular Technology*, vol. 64, no. 1, pp. 378–385, 2015.



Ricard Garra is a PhD student in the Cryptography and Graphs research group at University of Lleida, Spain. He received his B.S. and MSc. in Computer Engineering from University of Lleida in 2012 and 2014, respectively, and a MSc. in Computational & Software Techniques in Engineering from Cranfield University, United Kingdom, in 2014. His fields of interest include cryptography over elliptic and hyperelliptic curves.



Santi Martínez is a Postdoctoral Research Assistant in the Department of Mathematics at University of Lleida, Spain. He received his B.Sc. and M.Sc. in Computer Engineering from Rovira i Virgili University, Spain, in 2002 and 2004, respectively, and his Ph.D. in Engineering from University of Lleida in 2011. He has participated in several Spanish funded research projects and has authored over 15 publications. His research interests include order preserving encryption, RFID systems and elliptic curve cryptosystems.

Francesc Sebé is an Associate Professor in the Department of Mathematics at University of Lleida, Spain. He got his M.Sc. in Computer Engineering from Rovira i Virgili University, Spain, in 2001 and his Ph.D. in Telematics Engineering from the Technical University of Catalonia, Spain, in 2003. He has participated in several European and Spanish funded research projects. He has authored over 70 international publications. His fields of interest are cryptography and information privacy.