Parking Tickets for Privacy-Preserving Pay-by-Phone Parking*

Ricard Borges[b], Francesc Sebé[a,b]
(a) Center for Cybersecurity Research of Catalonia.
(b) Department of Mathematics. Universitat de Lleida.
C. Jaume II, 69, E-25001 Lleida, Catalonia, Spain
e-mail: {rborges,fsebe}@matematica.udl.cat

Abstract

Traditionally, the payment required for parking in regulated areas has been made through parking meters. In the last years, several applications which allow to perform these payments using a mobile device have appeared.

In this paper we propose a privacy-preserving pay-by-phone parking system offering the same privacy as the traditional paper-based method even assuming an internal attacker with full access to all the information managed by the system servers. Drivers'privacy is preserved without requiring them to trust any party. Furthermore, the system can tolerate that the mobile devices of drivers fall out of coverage while their cars are parked.

1 Introduction

In the late 1920s, Roger W. Babson filed several parking meter patents [1]. In most proposals, after inserting coins into a pay station, a paper ticket to be placed on the dashboard of the car is issued.

The massive deployment of smart devices has facilitated the development of applications allowing such payments to be performed through the mobile phone [2, 3, 4, 5]. Upon parking, the driver simply has to log into the mobile

*This document is a postprint version of the paper presented at the 18th Workshop on Privacy in the Electronic Society (WPES'19), November 11, 2019, London (United Kingdom). DOI: 10.1145/3338498.3358638

app and indicate the car license plate number, the area of the city she has parked in, and the expected duration. The amount to pay is then deducted from a pre-paid balance, or charged directly on driver's credit card.

Parking enforcement officers check the parking status of a car by typing its plate number in a mobile device which indicates whether a payment is in effect or not. This requires the presence of an online server, accessible from officers'devices, with access to the data allowing to determine the payment status of cars. This data is highly sensitive since it allows to infer private information about drivers like their work schedules, hobbies, or even health problems. To avoid unnecessary risks, the only information that should be managed by the system is that allowing a parking officer to check if an appropriate payment for a given parked car has been made or not.

Privacy is a key aspect in the design of secure systems and protocols involving the mobility of people and vehicles. That is the case for electronic transport tickets [6], pseudonym management in vehicular networks [7], management of parking space [8, 9], or vehicle location proof systems [10].

To avoid security breaches from inside the service provider, pay-by-phone parking systems should be designed by considering it a party which might try to infer information about drivers' parking habits. This excludes checking the payment status of a car just from its plate number. In that case the service provider could simply query the system periodically with a targeted plate number and get accurate information about the time periods that the traced car has been parked. Hence, the payment status of a car has to be determinable only from one-time pseudonyms which can only be obtained by a parking officer located close to the car[1]. This forces the need to include some kind of on-board device providing one-time pseudonyms when requested by a parking officer [11, 12].

1.1 Related Work

Nowadays there exist several pay-by-phone parking systems [2, 3, 4, 5], but none of them addresses the privacy of drivers. These applications collect and store accurate data about each parking operation so that the generation of reports about parking habits can be done in a straightforward manner.

Recent research works [11, 12] have proposed systems that consider the privacy of drivers. Both proposals share a similar system model. Pre-paid e-coins are used for anonymous payments. In addition, an RFID-enabled device is to be placed in cars and queried by parking officers when checking

[1] In this way the tracking of vehicles can not be done better than by patrolling the city for collecting information about parked cars.

the parking status of cars.

In [11], when a driver parks her car, an anonymous e-coin payment for the expected parking duration is made. That payment is linked to a random identifier which is stored in the on-board RFID device. When a parking officer checks a car, he queries its on-board device to get the random identifier which is then sent to the system server to determine whether a payment linked to it has been done. At that time, the officer can link the car license plate number to the current identifier, and from the data stored at the system servers, that identifier can be linked to the start and end times of the parking operation. Hence, the exact start and end times of the parking operation of that car are determined.

The proposal in [12] provides better privacy: the parking officer, when checking the parking status of a car, only obtains a boolean indicating whether a payment for the checked car is in effect or not. That proposal performs periodic micropayments for short-time intervals while the car is parked. A payment can only be linked to a plate number after querying the on-board device. In that case, the car can only be linked to the payment performed for the current short-time interval so that the start and end times keep secret.

The system proposed in [12] is currently the most complete privacy-preserving pay-by-phone parking system. Unfortunately, if the driver's mobile device could not perform some of the micropayments due to a lack of coverage, low battery or any other cause, the driver could be fined. Our proposal provides all the features of [12] and, additionaly, solves the mentioned drawback.

1.2 Contribution and Plan of this Paper

We propose a privacy-preserving pay-by-phone parking system in which, like in [12], payments are performed for short-duration time slots using pre-paid e-coins. In our proposal, a payment for each of the time slots composing the expected parking time is done at the beginning of the parking operation, eliminating the need to be connected all the time. Unused parking tickets can be revoked so that the driver recovers the money corresponding to unused time.

This first section has exposed the privacy concerns arising from the deployment of pay-by-phone parking systems. Next, Section 2 explains the system and adversary models together with the privacy requirements that the system should fulfill. After that, our proposal is detailed in Section 3, while Section 4 concludes the paper.

2 System and adversary models

In the proposed system, payments are performed for short-duration time intervals (5 or 10 min). As a result of paying for a given time slot, the driver gets a *parking ticket* for that slot. When a parking operation begins, the driver pays and gets a parking ticket for each of the time slots composing the expected parking time. Parking tickets are paid using pre-paid e-coins[2]. Unused tickets can later be revoked in advance.

This section describes the system and adversary models. After that, the privacy requirements are detailed.

2.1 System Model

Our system is composed of the following actors:

1. *Mobile application.* It is run on the mobile device of drivers. It allows to acquire pre-paid credit (in the form of e-coins), request parking tickets, revoke unused tickets, and provide a parking ticket when requested by a parking officer.

2. *System server.* It is an on-line platform accessed by the mobile application to manage parking operations, and by parking officers to check the payment status of cars.

3. *On-board RFID device.* It is placed inside cars and queried via RFID by a parking officer during an inspection.

4. *Parking officer.* He patrols regulated parking areas, queries the on-board RFID device of cars, and asks the system server about their parking status.

5. *Timestamp authority:* It issues timestamps upon request by the system server.

Use cases involving communication are enumerated next:

1. *Application set-up.* When a driver installs the app she is asked to introduce the car plate number and a credit card required to acquire pre-paid e-coins. Then, the app transmits the plate number and

[2]So as to preserve privacy, the employed e-coin system must be untraceable, like [13]. This excludes the use of transferable cryptocurrencies with a publicly available transaction history. In our proposal we have had to design an ad-hoc e-coin system able to deal with *valued* and *no-valued* e-coins.

the current time to the on-board device. The on-board device then generates an RSA key-pair and transmits the public key to the app. This communication must require authentication by the driver. It is performed via some short-range system like NFC or Bluetooth.

2. *E-coin request.* The app connects to the system server and requests for a certain amount of e-coins which are pre-paid via credit card. The price of each e-coin corresponds to the price of parking during a time slot. As it will be explained later, e-coins can be *valued* or *no-valued*[3].

3. *Parking ticket generation.* The app connects to the system server and gets and stores parking tickets after performing the corresponding anonymous payments in e-coins. This communication is done through an anonymous channel to avoid linking parking payments from IP addresses [14].

4. *Parking inspection.* A parking officer first queries the on-board device of a car which responds with a signed message containing the current time slot and its plate number. The officer forwards it to the system server which will connect to the corresponding mobile device via the *"Parking ticket request"* protocol. If the server gets a proper response, the officer will be notified. Otherwise, the car will be fined.

5. *Parking ticket request.* The system server employs a push notification service [15] to ask for a parking ticket to the app of a driver. If the mobile device is connected, the response is immediate. Otherwise the car will be fined. When connected again, if the mobile phone provides a parking ticket valid for the time the fine was issued, the fine will be canceled.

6. *Parking ticket revocation.* If a parking operation takes less time than expected, the driver can revoke unused parking tickets. The app connects to the server through an anonymous channel and revokes them receiving valued e-coins in exchange.

2.2 Adversary Model

The considered adversary model equals the one described in [12]:

1. The mobile application and the on-board device can not be corrupted.

[3]No-valued e-coins are used to issue dummy parking tickets which mask the expected duration of parking operations.

2. System servers and parking officers will follow the protocol steps as specified but may collaborate with an adversary by providing all the data they have access to.

From the service provider point of view, drivers are untrusted parties which might try to get parking time without paying for it.

2.3 Design Objectives

In this section we describe the design objectives. Privacy requirements are equivalent to those in [12]. An additional requeriment regarding system tolerance to mobile phones getting out of coverage has been added.

1. Parking tickets can only be linked to a car plate number as a result of a parking status checking performed by a parking officer located close to the car.

2. The information provided by a driver during a parking inspection only allows to determine whether a parking ticket for the current time does exist or not.

3. During a parking operation, the mobile app only needs an Internet connection at the beginning and at the moment of revoking unused parking tickets.

3 Our proposal

This section describes the procedures that compose the proposed system.

1. *System parameters generation.* This procedure must be run by the service provider before deploying the service:

   (a) Generate an RSA [16] key pair for the system server. Let V_S be the private key (only known to the server) and let P_S be the public key.

   (b) Generate an RSA key pair for the timestamp server. Let V_{TSA} be the private key (only known to the timestamp server) and let P_{TSA} be the public key.

   (c) Generate a tuple (p,q,g) of DSA [17] public set-up parameters which are stored at the system server.

Rationale. The system server will use its key pair to (blindly) sign e-coins and parking tickets during their issuance preventing, in this way, malicious drivers from being able to forge them. Timestamps will be issued to avoid disputes regarding the revocation status of parking tickets.

2. *Application setup.* A driver must install the app into her mobile device and enter the car license plate number and a credit card number (necessary to pay for the purchased e-coins). The app will then connect to the on-board device and send the plate number and the current time. The on-board device then sets its internal clock and generates an RSA key pair: V_D (private) and P_D (public). The public key P_D is transmitted to the mobile app.

   Rationale. The on-board device key pair will be used to generate signed messages which are returned to parking officers after being queried by them. Possession of such a signed message proves that a parking officer was located close to the car when a parking inspection took place.

3. *E-coin request.* The app includes a wallet which stores e-coins that are generated through a protocol run between the app and the system server. A *valued* e-coin has a price while *no-valued* ones are free of charge.

   (a) A *valued* e-coin is generated as follows:

      i. The mobile app performs a credit card payment for the requested e-coin[4].
      ii. The app generates a DSA key pair V_C/P_C over the (p, q, g) parameters.
      iii. The app generates an *even* number $x \in [0, (q-1)/2)]$.
      iv. The app computes $H(P_C||g^x)$ and asks the server to compute an RSA blind signature on it.
      v. The app gets $Sign_S(H(P_C||g^x))$ as a result.

   (b) A *no-valued* e-coin is generated as follows:

      i. The app generates N DSA key pairs $\{V_{Ci}/P_{Ci}\}_{0 \leq i < N}$ and N *odd* numbers $\{x_i\}_{0 \leq i < N} \in [0, (q-1)/2)]$.
      ii. Then, the app computes $\{H_i = H(P_{Ci}||g^{x_i})\}_{0 \leq i < N}$ and blinds each H_i with a different blinding factor r_i. All the N blinded hash digests are sent to the server.

[4]E-coins will usually be generated in batch so that just a single credit card payment for the overall amount will be done.

iii. The server randomly chooses an index $j \in [0, N-1]$ and sends it to the app.

iv. For each $i \neq j$, the app sends r_i, x_i and P_{Ci} to the server.

v. The server unblinds the $N-1$ hash digests H_i and checks whether each H_i equals $H(P_{Ci}||g^{x_i})$. It also verifies that each x_i is odd and falls in the $[0, (q-1)/2)]$ range.

vi. If all the checkings are satisfied, the server blindly signs the remaining blinded hash and returns the result to the app.

vii. The app gets $Sign_S(H(P_{Cj}||g^{x_j}))$ per result.

Rationale. The difference between valued and no-valued e-coins is the parity of x. The system server, before issuing a no-valued e-coin must check that the corresponding x is odd through the described cut-and-choose technique. E-coins are blindly signed so that their issuance and spending can not be related. Regardless of being valued or no-valued, for each issued e-coin, the app stores a tuple:

$$\{Sign_S(H(P_C||g^x)), V_C, x\}.$$

4. *Parking ticket generation.* The driver has to issue a (valued) parking ticket for each of the time slots that compose the expected parking period. To hide its duration, the app always requests the same amount, R, of tickets (embracing the maximum allowed parking time). The tickets for slots in which the driver expects to be parked are paid with valued e-coins. The remaining ones are issued with no-valued ones so that no-valued dummy tickets are generated. This procedure is run through an anonymous channel:

   (a) The app takes a (valued or no-valued) e-coin

   $$\{Sign_S(H(P_C||g^x)), V_C, x\},$$

   and spends it by computing $P_C = g^{V_C}$ and $G = g^x$, and sending $Sign_S(H(P_C||G))$, P_C, and G to the server. Then, employing the secret key V_C the app signs a bitstring representing the current time and sends the resulting signature $Sign_C(H(CurrentTime))$ to the server.

   (b) The server verifies the RSA signature $Sign_S(H(P_C||G))$ using its public key P_S, the signature $Sign_C(H(CurrentTime))$ under the e-coin public key P_C and checks that the e-coin has not been spent before.

(c) The app determines the current time slot, $Slot$. Then it generates N random keys $\{K_i\}_{0\leq i<N}$, and computes the set $\{ID_{Slot i} = HMAC_{K_i}(Slot||License)\}_{0\leq i<N}$. It also generates N even numbers $\{x_{ri}\}_{0\leq i<N} \in [0,(q-1)/2)]$ and computes $\{G'_i = g^{x'_i} = g^{x+x_{ri}}\}_{0\leq i<N}$.

(d) The app next computes the following N hash digests $\{H(Slot||ID_{Slot i}||G'_i)\}_{0\leq i<N}$, blinds each of them with a different blinding factor r_i, and sends the blinded results to the server.

(e) The server randomly chooses an index $j \in [0, N-1]$ and sends it to the app.

(f) For each $i \neq j$, the app sends r_i, x_{ri}, $Slot$ and $ID_{Slot i}$ to the server.

(g) The server unblinds the $N-1$ hash digests H_i, computes $G'_i = G \cdot g^{x_{ri}}$, and checks whether each H_i is equal to $H(Slot||ID_{Slot i}||G'_i)$. It also verifies that each x_{ri} is even and falls in the $[0,(q-1)/2)]$ range, and checks that $Slot$ corresponds to a future time slot.

(h) If all the checkings are satisfied, the server blindly signs the remaining blinded hash and returns the result to the app. Also, the server stores:

$$\{P_C, G, Sign_S(H(P_C||G)), CurrentTime,$$

$$Sign_C(H(CurrentTime))\}.$$

After running this process, the app gets a signature on the ticket, namely $Sign_S(H(Slot||ID_{Slot}||g^{x'}))$. If $x' \in [0, q-1]$ is even, the ticket is valued. For each *valued* ticket, the app stores:

$$\{Sign_S(H(Slot||ID_{Slot}||g^{x'})), Slot, K, x'\}.$$

Rationale. A parking ticket is valued or not depending on the parity of x'. The system server must check that x (in the e-coin) and x' (in the ticket) have the same parity. Since $x' = x + x_{r_j}$, this is achieved by checking that x_{r_j} is even by means of the described cut-and-choose technique.

5. *Parking inspection.* A parking officer approaches to the car and queries its on-board device.

(a) The on-board device, from its internal clock, determines the current time slot, $Slot$, and sends it to the parking officer together with the car plate number, $License$, and the signature $Sign_D(H(Slot||License))$.

(b) The officer checks the received $Slot$ and $License$ values and sends them together with $Sign_D(H(Slot||License))$ to the system server.

(c) The server sends a push message to the app associated to $License$ and waits for a valid parking ticket (see the *"Parking ticket request"* procedure) during some time.

(d) If the app does not properly respond to the request, the car is fined and its driver receives a telematic notification.

Rationale. Possession of $Sign_D(H(Slot||License))$ proves that the parking officer is located close to the car.

6. *Parking ticket request.* When the app is requested to prove a pending inspection, the following procedure is run:

   (a) The app:

      i. Connects to the server and asks for the signature computed by the on-board device (during the *"Parking inspection"* execution) over $Slot$ and $License$.

      ii. Verifies the signature $Sign_D(H(Slot||License))$.

      iii. If the previous verification is successful, it sends the ticket $Sign_S(H(Slot||ID_{Slot}||g^{x'}))$, $Slot$, ID_{Slot}, K, and x' to the server.

   (b) The system server:

      i. Checks the signature $Sign_S(H(Slot||ID_{Slot}||g^{x'}))$ over $Slot$, ID_{Slot}, and $g^{x'}$ under the server public key P_S;

      ii. Verifies that ID_{Slot} equals $HMAC_K(Slot||License)$

      iii. Checks that $x' \in [0, q-1]$ is even and has not been previously revoked. In such a case, the timestamp issued during the revocation, $Timestamp_{TSA}(x')$, is returned (see the *"Parking ticket revocation"* procedure).

   (c) If all the previous checkings are satisfied, the parking officer is informed about the validity of the provided ticket.

If the app can not provide a valid ticket, the driver is fined. If the driver paid for parking but her phone is out of coverage, she will also be fined. In that case, the fine will be canceled when the mobile device provides the parking ticket after getting connected again.

7. *Parking ticket revocation.* If the driver removes the car before than expected, she can revoke unused valued tickets and recover the paid amount. The app connects to the server through an anonymous channel, and, for each valued parking ticket:

   (a) The mobile app sends $Sign_S(H(Slot||ID_{Slot}||g^{x'}))$, $Slot$, ID_{Slot} and x'.

   (b) The system server:

      i. Validates the signature $Sign_S(H(Slot||ID_{Slot}||g^{x'}))$ over $Slot$, ID_{Slot}, and $g^{x'}$ under the public key P_S.

      ii. Checks that $Slot$ corresponds to a future time slot.

      iii. Checks that x' is even and has not been revoked before. In such a case, its $Timestamp_{TSA}(x')$ would be returned as a proof.

   (c) If the parking ticket to be revoked is valid it requests the TSA to timestamp x'. It stores x' and $TimeStamp_{TSA}(x')$ in a database.

   (d) Finally, a new valued e-coin is issued and stored by the app wallet (by running the *"E-coin request"* procedure).

   Rationale. Before revoking a ticket, the system server checks that it is valued (x' is even). The timestamp on x' will serve, in case of dispute, as a proof of its revoked status.

4 Conclusion

In this paper, a pay-by-phone parking system which guarantees the privacy of drivers' parking habits has been proposed. Drivers pay for a set of parking tickets at the beginning of a parking operation and, if the parking operation takes less time than initially expected, they can revoke unused tickets.

Acknowledgment

This work was partially funded by the Spanish Ministry of Science, Innovation and Universities under Project MTM2017-83271-R.

References

[1] R. W. Babson, "Automatic parking meter," *US Patent 1973275A*, 1934.

[2] "Pango mobile parking," 2019.

[3] "Parkmobile," 2019.

[4] "Parkright," 2019.

[5] "Paybyphone," 2019.

[6] M. Mut-Puigserver, M. Payeras-Capellà, J. Ferrer-Gomila, and J. Castellà-Roca, "A survey of electronic ticketing applied to transport," *Computers & Security*, vol. 31, pp. 925–939, 2012.

[7] J. Petit, F. Schaub, M. Feiri, and F. Kargl, "Pseudonym schemes in vehicular networks: a survey," *IEEE Communication Surveys & Tutorials*, vol. 17, no. 1, pp. 228–255, 2015.

[8] G. Yan, W. Yang, D. Rawat, and S. Olariu, "Smartparking: a secure and intelligent parking system," *IEEE Intelligent Transportation Systems Magazine*, vol. 3, no. 1, pp. 18–30, 2011.

[9] C. Huang, R. Lu, X. Lin, and X. Shen, "Secure automated valet parking: a privacy-preserving reservation scheme for autonomous vehicles," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 11169–11180, 2018.

[10] Y. Zhang, 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.

[11] P. Pérez-Martínez, *Contributions to Privacy Protection for Ubiquitous Computing*. PhD thesis, Universitat Rovira i Virgili, 2015.

[12] R. Garra, S. Martínez, and F. Sebé, "A privacy-preserving pay-by-phone parking system," *IEEE Transactions on vehicular technology*, vol. 66, no. 7, pp. 5697–5706, 2017.

[13] D. Chaum, "Blind signatures for untraceable payments," in *Proc. of Crypto'83*, 1983.

[14] "Tor project," 2019.

[15] R. Boyer and K. Mew, *Android Application Development Cookbook*. Packt Publishing, 2016.

[16] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, pp. 120–126, 1978.

[17] NIST, "Digital signature standard," tech. rep., National Institute of Standards and Technology, 2013.