This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2014.2361016, IEEE Internet of Things Journal

Privacy-Preserving Cooperative Route Planning

Martin Florian, Sören Finster, Ingmar Baumgart Institute of Telematics, Karlsruhe Institute of Technology (KIT)
76131 Karlsruhe, Germany
Email: {florian,finster,baumgart}@kit.edu

***Abstract*—Today's street traffic is still largely inefficient. Overburdened roads lead to congestions, accidents and unnecessary pollution. The increasing interconnection of traffic participants into the *Internet of Vehicles* (IoV) has tremendous potential for improving this issue. *Cooperative route planning*, for example, is a concept for optimizing vehicular routing on a global scale by gathering data about planned routes from interconnected vehicles. As in other IoV applications, the benefits of such a system come at the cost of an increased privacy risk for participating users. Published routes include both the current and the planned future locations of drivers and passengers - all highly sensitive pieces of information. In the scope of this paper, we demonstrate how cooperative route planning can be realized with strong privacy guarantees without significant cuts in utility or cost. According to our knowledge, this is the first work to consider this issue. We propose a scheme by which vehicles can publish their intent to pass at specific waypoints at approximate times in an anonymous fashion. While providing complete unlinkability of published intentions to individual users, our scheme is protected against abuse, with misbehaving (i.e., lying) users quickly losing their right to participate.**

I. Introduction

The distribution of vehicular traffic today is still largely inefficient. Overburdened roads lead to congestions, traffic accidents and increased pollution due to stop and go. Adaptive route planning based on traffic sensing is a widely accepted measure for improving things. Vehicles receiving traffic updates can adapt their routes accordingly, thus reaching their destinations quicker and contributing to the overall traffic flow. Most currently deployed systems restrict themselves to estimations about the current traffic situation. Predictions about the future, if at all, are made only based on the current state and historic data. The growing interconnection of vehicles and infrastructure, culminating in the *Internet of Vehicles* (IoV), allows for more advanced forms of cooperation in vehicular route planning. Information about planned routes can be shared by vehicles on the road, leading to significantly more precise traffic predictions. This allows for faster routes to be found and results in a significantly improved traffic flow [6], [9]. In the following, we will refer to the approach of sharing planned routes and integrating the plans of others into route planning decisions as *cooperative route planning*.

As in other services building on top of the IoV, a central challenge in cooperative route planning is the protection of user privacy. A user's location, driving habits and intended destination are highly sensitive pieces of information. With only an indirect benefit from publishing their intended route, users might choose against participating in cooperative route planning if they face the danger of potential privacy loss. It is thus desirable to grant full anonymity to all cooperative route planning participants, as well as unlinkability between their actions. However, this raises the question of trust: how to ensure that a published plan is trustworthy if it cannot be linked back to a specific user in any way? How to prevent cheating and ensure that malicious users are excluded from the system?

To tackle these questions, we propose a system based on the concept of *promise coins* (PCs) - cryptographic constructs related to Chaum's electronic cash [2]. PCs are used to prove the trustworthiness of promises without giving up user anonymity. New PCs are issued only to users that fulfill their promises, thus excluding non-cooperative or malicious users quickly. We provide a detailed description and analysis of our proposal and evaluate possible performance bottlenecks to prove its practicability. According to our knowledge, our solution is the first to realize both strong privacy and abuse-resistance in cooperative route planning.

II. High-Level Overview

We start by giving an abstract overview over our scenario and the specific solution we propose. We introduce the *cooperative route planning* concept, our *promise coin* construction and, lastly, its application to the problem of realizing abuse-resistant cooperative route planning with strong privacy guarantees.

A. Cooperative Route Planning

In the following, we will motivate the *cooperative route planning* concept based on a typical use-case. Consider an ideal route planning system and the following scenario: Before a user starts his trip, he enters his destination into his navigation device[1]. His navigation device receives both updates about the current traffic situation as well as precise predictions about future developments. In this way, it can propose a route, a start time and times and locations for possible coffee breaks, so that the user wastes as little time and fuel as possible. As a side-effect, by ensuring a fast and uncongested journey for the user, his navigation device also contributes towards improving the global traffic flow for all traffic participants.

The effectiveness of the navigation device's planning largely depends on the quality of the data it receives. For the current traffic state, good estimates can be obtained by gathering data

[1]Automated route planning is even more interesting when considering self-driving cars which, unlike human drivers, cannot function without it. Also, they always stick to the route proposed by their route planning component.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2014.2361016, IEEE Internet of Things Journal

from road side infrastructure or speed measurements from vehicular fleets. For calculating the future traffic state, on the other hand, input from all other vehicles on the road is desirable for deriving an accurate forecast. If all vehicles on the road publish their intended route and consider the published routes of others, significant improvements in overall traffic flow can be achieved [6], [9]. We use the term *cooperative route planning* for describing this approach.

In its pure form, cooperative route planning has a strong altruistic element - by publishing his plans, a user contributes to the common good. Additional incentives can easily be introduced into such a system as well. For example, participating users can receive benefits, like road toll discounts or the right to use a reserved lane, if they publish their plans beforehand. In any case, insufficient trust in the privacy provided by a cooperative route planning system can hamper its deployment. Users may weigh their own privacy higher than the common good or the benefits a service operator may offer. Additionally, the centralized collection of large amounts of sensitive data is undesirable in the context of maintaining democratic societies.

In the scope of this paper, we show that a cooperative route planning system can be realized with strong privacy guarantees, while minimizing its vulnerability to abuse and without incurring significant additional costs.

B. Promise Coins

The main challenge when providing strong privacy for users in cooperative route planning systems is to ensure that malicious users cannot disturb the functionality of the system. A single malicious user might, for example, repeatedly publish made-up routes, making individual road segments appear overburdened and thus greatly altering the flow of traffic. Given a simple free for all system with full user anonymity, it is impossible to exclude such users even if their misbehavior can be identified. Consequently, in a completely anonymous system of peers, the promises of other participants cannot be fully trusted, which leads to a degraded system utility. On the other hand, user anonymity and unlinkability of user actions is highly desirable from a privacy standpoint.

To answer this tension, we propose a scheme based on the concept of *promise coins* (PCs) - cryptographic constructs related to Chaum's electronic cash [2]. In electronic cash, financial authorities issue digital bills to users using *blind signatures*, i.e., without being able to link these bills to the receiving users later on.

In our proposal, a pool of promise coins is issued to each user upon authentication at a central authority, using a blind signature scheme. In the following, we will refer to this central authority as the *Promise Authority* (PA). PCs are used to make anonymously published *promises* about a user's planed route trustworthy. From a high-level view, a user publishes a promise by *paying* the PA one PC for it. In exchange, he receives a *promise token* from the PA. Once the user proves that he has fulfilled his promise, or if he revokes the promise in a timely manner, he can redeem the promise token and *receive a new PC* from the system. Depending on the specific deployment scenario, the user might additionally receive benefits when

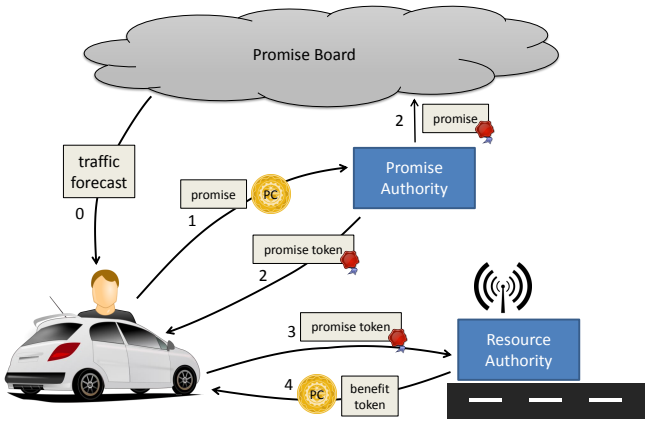

Fig. 1. High level overview.

redeeming a promise token, like a discount on the road toll for the current road segment. Promises can also be treated as reservations. For example, only users with valid promise tokens might be allowed to pass a given tunnel or bridge. Honest users will never run out of PCs and will never need to perform the identity-coupled PC issuing protocol with the PA again. Uncooperative users, on the other hand, will run out of PCs quickly and can be denied new PCs by the PA, effectively banning them from the system. Using blind signatures for the issuing of PCs, we ensure that they are completely unlikable to specific users or to other PCs of a user.

C. Cooperative Route Planning with Promise Coins

When calculating routes, user devices take traffic updates from the *Promise Board* (PB) into account. The PB is the publicly accessible database of all currently relevant promises. Thus, traffic updates from the PB include detailed traffic forecasts based on the published promises of other participants. Once a route has been found, the user (respectively its navigation device) identifies parts of its route for which promises can be made. Specifically, it identifies *resources* (road segments, tunnels, etc.) along its planned route, for which a *Resource Authority* (RA) has been deployed. For each of these resources, it then formulates a *promise* containing a resource identifier and a time frame (e.g., a 15 minute timeslice) during which it expects the resource to be passed. Using an anonymous channel, this promise is transmitted to the *Promise Authority* (PA) together with a valid, previously unspent promise coin. After validation of both the promise and the PC, the PA commits the promise to the PB and returns a *promise token* (PT) to the user. Once the user reaches the resource region, he can show the promise token to the RA. If he arrived at the region within the promised time frame, the RA will cash in the promise token and issue a new promise coin to the user.

Fig. 1 gives a high-level overview over these steps. Namely, the diagram depicts the distribution of traffic updates (0), the publication of a promise (1), the commitment of a promise to the PB and the issuing of a promise token (2), the promise token redemption request at the responsible RA (3) and finally the issuing of a new PC upon the successful redemption of a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2014.2361016, IEEE Internet of Things Journal

promise token (4). In the last step, the optional granting of a benefit is shown as well, in the form of issuing a *benefit token*.

While we often use the term user when describing our approach, the actual work in a practical implementation will be performed by the user's navigation device or self-driving car. The details of our scheme can easily be hidden from drivers and passengers, making our system appear as a regular route planning application.

III. Design

Following the high-level overview given in Sec. II, we now present a detailed description of our proposal. We first introduce all employed entities and data objects. Following that, we describe the specific mechanisms composing our approach. An overview of all cryptographic certificates and data objects involved in our approach can be found in Tab. I.

A. Entities

1) User: We use the term *user* interchangeably for both an actual human driver using our proposed system as well as any device or piece of software acting on behalf of such a driver. Standard automated navigation devices and fully automated self-driving vehicles are both included in this definition. In a practical implementation, all details and mechanisms comprising our proposed system can easily be hidden from actual human users. In the spirit of the IoV, we assume vehicles to be equipped with both cellular (e.g., LTE) and short range radio (e.g., IEEE 802.11p) communication interfaces.

2) Promise Authority (PA): The *Promise Authority* (PA) is the main trust anchor in our proposed system. It maintains the well-known *Promise Authority Certificate* (PAC). The PAC is used for establishing trusted communication channels with the PA, signing promise tokens and validating actions concerning the state of the PB (e.g., the revocation of promises). The PA additionally maintains the *Coin Issuing Certificate* (CIC). The CIC is only used for blindly signing PCs. A PC is only valid if it includes a signature by a currently valid CIC[2].

The PA can be maintained by a public traffic authority or a private company providing the cooperative route planning service. In the scope of this paper, for simplicity, we focus on a setup featuring one PA. In practice, the deployment of multiple PAs might be interesting for achieving higher fault tolerance and a decentralization of control.

3) Resource Authorities (RAs): A *Resource Authority* (RA) is an entity responsible for a specific resource, e.g., a road segment, bridge or tunnel. In order for a user to be able to make promises about a resource, a RA needs to be deployed for that resource. RAs are equipped with a *Resource Authority Certificate* (RAC) signed by the PA. RACs are used for authentication and their public part is used by the PA for generating promise tokens. All RAs, their resources and their RACs are publicly known. Information about RAs can be distributed, for example, by the PB.

The main function of a RA is the verification of the fulfillment of promises. Thus, a RA needs to be both reachable for users passing its resource and be able to verify that users are really passing it. For satisfying both of these requirements, we envision RAs to be equipped with short range radio interfaces as widely discussed in the vehicular networking community [12]. Radio-equipped road side units (RSUs) are a central element of the IoV and are expected to find major deployment, e.g., for relaying safety messages and sensor readings. We propose to implement RAs in such a way that they have access to one or multiple RSUs scattered around the respective resource managed by them. Other existing infrastructure like toll stations can be leveraged by RAs as well. However, we assume that RAs have no access to dedicated vehicle identification equipment like license-plate scanners. While the anonymous publication of routes using our approach is still possible with RAs equipped in this way, the routes vehicles have already taken will not be easily anonymizable.

Upon verifying that a promise has been fulfilled, RAs cash in the supplied promise token and hand out a new PC to the user. This step, again, involves only the RA and the user. In general, none of the actions performed by a RA requires any communication with the PA, PB or other RAs. Thus, as an important detail, RAs can safely be realized in an "offline" manner without any form of long-range (i.e., Internet) connectivity[3].

4) Promise Board: The *Promise Board* (PB) is a publicly accessible database for promises. The main function of the PB is to keep track of published promises and to make this information public to all interested parties. Information from the PB can be used for making routing decisions, by checking how much demand for a resource (road segment) is expected in the future. Promises and promise revocations are pushed to the PB by the PA after they have been validated.

The PB can be realized as a centralized service, maintained by the service operator. Alternatively, it can also be realized in a distributed manner using a distributed data structure like a distributed hash table (DHT). Peer2PeerTIS [14], for example, is a DHT optimized for storing traffic-related data that might be extended for supporting time dependent traffic estimations based on published promises.

B. Objects

1) Promise: In the context of this paper, a *promise* is a public statement of intent from an anonymous user to make use of a specific resource within a specific timeslice. So, a promise P has the form:

$$P := (\text{resource identifier}, \text{timeslice})$$

In a cooperative route planning context, a promise can have a semantic like "I will be passing the Gotthard tunnel going south, between 11:00 and 11:30 today.". Depending on the context and the type of the resource, a promise can also

[2]In a practical deployment, CICs will likely need to be rotated after certain periods to maintain the scalability of PC double spending detection. Such rotations are also beneficial for improving the security and the overall manageability of the system and PC supply [2]. A CIC rotation can happen seamlessly to users, with the PA offering to exchange PCs signed with the old CIC for new ones.

[3]However, with RSUs connected to the Internet, individual RAs can also be realized as cloud-based services, thus further reducing deployment costs.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2014.2361016, IEEE Internet of Things Journal

TABLE I
USED CERTIFICATES AND DATA OBJECTS.

Object	Authentication	Function
PA Certificate (PAC)	external, trust anchor	secure connections to PA, trust anchor
Coin Issuing Certificate (CIC)	signed with PAC	blind signing of PCs
Promise Coin (PC)	signed with CIC	authenticating promises
Promise	by spending a PC, later signed with PAC	publishing future intent
Promise Token (PT)	signed with PAC	redeemed for a new PC upon promise fulfillment
Certificate (RAC)	signed with PAC	RA authentication, generating (public part) and cashing in PTs
User ID	external	requesting first batch of PCs

be treated as a reservation, i.e., only vehicles that made a reservation might be allowed to pass the Gotthard tunnel at a specific high peak time. In a vehicular traffic context, resources represent short road segments with a length of up to several kilometers. For sharing whole routes, users publish multiple promises. Given a sufficient deployment of RAs, a user can publish promises covering its entire planned route.

2) Promise Coins: A *promise coin* (PC) is a cryptographic construct consisting of a random *coin ID* and a signature by the PA using the CIC. A promise coin X can be formally defined as:

$$\mathrm{PC}_X := (\mathrm{coinID}_X, \mathrm{sig}_{\mathrm{CIC}}(\mathrm{coinID}_X))$$

PCs do not include a value field as they all have the same semantic: each PC can be used for publishing exactly one promise. This offers the benefit that the blind signature protocol can be completed in one round and the user needs to generate only one coin candidate. Since all PCs have the same "value", the PA can safely sign any blinded coin ID it receives.

3) Promise Token: Promise tokens (PTs) are generated upon the publication of a promise. They enable users to receive new PCs and benefits upon promise fulfillment. Promise tokens are composed of a promise, an encrypted blinded CIC signature for a new PC and a PAC signature. The blinded PC signature is encrypted in such a way that both the RA and PA can decrypt it in the case of promise fulfillment or promise revocation. We denote this encryption as $\mathrm{enc}_{\mathrm{RAC,PAC}}()$. In its simplest implementation, $\mathrm{enc}_{\mathrm{RAC,PAC}}()$ can be realized by concatenating the results of $\mathrm{enc}_{\mathrm{RAC}}()$ and $\mathrm{enc}_{\mathrm{PAC}}()$. The promise token $\mathrm{PT}_{P,Y}$ for a promise P and a new PC candidate Y can now be formally defined as (using a temporary value a):

$$a := (P, \mathrm{enc}_{\mathrm{RAC,PAC}}(\mathrm{sig}_{\mathrm{CIC}}(\mathrm{blind}(\mathrm{coinID}_Y))))$$

$$\mathrm{PT}_{P,Y} := (a, \mathrm{sig}_{\mathrm{PAC}}(a))$$

The specific mechanisms involved in constructing a promise token will be introduced in greater detail in the remainder of this section.

C. Mechanisms

1) Establishment of an Anonymous Secure Communication Channel: In order to avoid linkability of user actions based on communication metadata like communication addresses, an anonymous communication channel must be established for non-local communication (i.e., all communication over the cellular network). Examples for communication that needs to be protected in this way includes the publishing and revoking of promises at the PA and, depending on the implementation, the querying of the PB for traffic forecast data. Here, we assume the use of an existing anonymous communication service, namely the *Tor* network [8]. The Tor network is openly accessible and has a large bandwidth capacity. In comparison to regular mix networks, it offers low communication latencies and has the additional benefit that it establishes circuits, i.e., stateful paths through the network that provide bidirectional communication channels. In our scenario, this enables for data to be sent back to a user via the same channel he used for making his request. For maximum unlinkability, we require that circuits to the PA are used for only one action involving the use of a PC and discarded afterwards. Thus, a user establishes a new communication channel for each promise he publishes.

We also require all communication channels used in our system to be secure, in the sense that man-in-the-middle attacks on the communication channels are impossible. Since all non-user entities in our system are equipped with verifiable cryptographic certificates and no direct user to user communication is required, the establishment of a secure connection over a Tor circuit is straightforward using, for example, TLS/SSL [7].

The specific steps a user needs to take for forming a non-local anonymous and secure communication channel are thus the following:

1) Establishing a Tor circuit with the destination.
2) Establishing a secure connection over the Tor circuit, e.g., using TLS/SSL.

For local communication, i.e., communication with RAs over short range radio, we assume that the user is not identifiable based on his communication address (which he can change) or other characteristics (e.g., his license plate, see Sec. III-A3). Thus, the establishment of a secure connection is sufficient in this case.

2) Initial Coin Generation: During initial coin generation, a user receives a batch of PCs upon identification at the PA. For users that are honest and thus never run out of PCs, this needs to be performed only once, namely when the user starts using the system for the first time. The specific mechanism is composed of the following steps:

1) For each new promise coin X that the user wants to request from the PA, he generates a new random coin ID.

$$\mathrm{coinID}_X := \mathrm{random}()$$

2) The coin ID is blinded and sent to the PA, together with a proof of the user's identity. Such a proof can easily be realized by equipping navigation devices with

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2014.2361016, IEEE Internet of Things Journal

certificates tied to the user's identity.

$$\text{PC request} := (\text{userID}, \text{blind}(\text{coinID}_X))$$

3) If the PA finds the user to be eligible for another PC, it answers with a CIC signature on the blinded coin ID.

$$\text{PC reply} := \text{sig}_{\text{CIC}}(\text{blind}(\text{coinID}_X))$$

4) Using this signature and information about the applied blinding, the user can construct a signature for the original unblinded coin ID and thus assemble a new PC.

$$\text{PC}_X := (\text{coinID}_X, \text{unblind}(\text{sig}_{\text{CIC}}(\text{blind}(\text{coinID}_X))))$$

A straightforward, RSA-based implementation of a blind signature scheme, i.e., of the $\text{blind}()$, $\text{unblind}()$ and $\text{sig}_{\text{CIC}}()$ functions, is described in the Appendix. Due to the blinding function used by the user, the PA never learns the original coin ID of the PC it is signing. Thus, it cannot link the resulting PC back to the user. The PA keeps track of the number of PCs issued to individual users upon user authentication. In this way, cheating users can be identified and denied new PCs.

3) Publishing a Promise: The process of publishing a promise P is composed of an exchange between the user and the PA. It can be subdivided into the following steps:

1) The user establishes an anonymous secure communication channel with the PA.
2) The user generates a new random coin ID. This will be the base for the PC he will get back upon promise fulfillment.

$$\text{coinID}_Y := \text{random}()$$

3) The user sends one of his PCs, his promise P, and the blinded new coin ID to the PA over the anonymous channel. He also stores the unblinded value of coinID_Y so that he can construct PC_Y at a later time.

$$\text{promise request} := (P, \text{PC}_X, \text{blind}(\text{coinID}_Y))$$

4) The PA checks the validity of the promise and of the used PC. Specifically, it verifies the soundness of the promise and the validity of the supplied PC's signature and checks that no PC with the same coin ID has already been spent before.
5) If the PA considers both the promise and the PC valid, it signs the promise and sends it to the PB.
6) The PA furthermore proceeds to generate a promise token for P and the new promise coin Y. It signs $\text{blind}(\text{coinID}_Y)$ and encrypts it in such a way that both the RA responsible for the resource in P and the PA itself can decrypt it. This is easily done using the public keys from the respective RAC and the PAC by using an asymmetric encryption scheme. Let a be the intermediate result of these steps.

$$a := (P, \text{enc}_{\text{RAC,PAC}}(\text{sig}_{\text{CIC}}(\text{blind}(\text{coinID}_Y))))$$

Using $\text{enc}_{\text{RAC,PAC}}()$ instead of only $\text{enc}_{\text{RAC}}()$ is necessary for being able to redeem the token in the case of a revocation of P or a failure of the RA. The PA finishes the construction of a promise token for P and Y by signing a:

$$\text{PT}_{P,Y} := (a, \text{sig}_{\text{PAC}}(a))$$

7) The resulting promise token is sent back to the user via the same communication channel initiated by the user.

$$\text{promise acknowledgement} := \text{PT}_{P,Y}$$

4) Promise Fulfillment and Promise Token Redemption: RAs are responsible for verifying the fulfillment of promises and cashing in promise tokens. For a user that has published a promise P and has arrived at the resource mentioned in P[4] within the timeslice mentioned in P (in other words, is fulfilling P), the specific steps are the following:

1) The user contacts the responsible RA using short range radio and establishes a secure local communication channel with it using its RAC.
2) The user transmits the promise token $\text{PT}_{P,Y}$ it received from the PA upon the publication of P.

$$\text{fulfillment request} := \text{PT}_{P,Y}$$

3) The RA verifies the PAC signature of the promise token. If it is valid, it extracts P from the promise token and determines if the user has fulfilled it. Most importantly, it verifies whether P concerns its own managed resource, whether the user is indeed within the limits of this resource and whether the time commitment noted in P has been held.
4) Upon validating that P was indeed fulfilled, the RA decrypts the blinded PC signature found in $\text{PT}_{P,Y}$ and sends it back to the user.

$$\text{fulfillment acknowledgement} := \text{sig}_{\text{CIC}}(\text{blind}(\text{coinID}_Y))$$

5) Since the user knows both the original unblinded coin ID and the employed blinding function, the receipt of the blinded CIC signature is equivalent to receiving a new PC.

$$\text{PC}_Y := (\text{coinID}_Y, \text{unblind}(\text{sig}_{\text{CIC}}(\text{blind}(\text{coinID}_Y))))$$

Given a good placement of the RA's communication infrastructure (e.g., if it is distributed over several RSUs covering the resource area), the proof that a user is indeed within the limits of a resource is implicitly given when communicating over short range radio.

5) Promise Revocation: Promise revocations are necessary if a user changes his route or finds out that he cannot reach the resource mentioned in his promise in time. The revocation mechanism is analogous to the promise fulfillment mechanism, with the main exception that it is performed between user and PA and that not promise fulfillment is checked, but whether the revocation is early enough to be considered valid. Upon a successful revocation, the PA decrypts the PC signature contained in the supplied promise token and sends an update to the PB.

[4]The arrival at a specific resource can easily be detected by users using GPS or short-range radio beacons from the RA.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2014.2361016, IEEE Internet of Things Journal

In order to avoid the exploitation of the revocation mechanism by malicious users, e.g., the deliberate publication of false promises and their latter revocation, revocations can additionally be penalized by the PA. Since PCs are atomic, penalties can only be realized probabilistically. Depending on the specific application scenario, the PA might choose, with a probability p, to not issue a new PC after a promise revocation. The probability p is then the revocation penalty. On a sitenote, if offline RAs are used, it is also possible for a user to both revoke a promise and later fulfill it at the respective RA. However, this leads to no benefit for the user. The impact of such behaviour on traffic prediction is comparable to the impact of a traffic participant not equipped with cooperative route planning capabilities. Inconsistencies in the overall traffic prediction are not possible, as RAs are only responsible for verifying promises and do not participate in traffic prediction.

IV. Analysis

In the following, we present an analysis of our scheme. We formulate a system model and, based on it, explore the security of our approach against abuse. We then discuss possible sources of privacy loss.

A. System Model

For analysis, we assume an implementation of our proposed system with one dedicated PA, one PB and multiple RAs associated with different resources. The PA, PB and RAs are all assumed to be run by the same organizational entity, e.g., the local traffic authority or a private company. In addition to these fixed entities, we assume the existence of a large number of honest users participating in the system. While it is not necessary that all vehicles on the road contribute by publishing promises (with knowledge about the usage ratio, interpolations can be made), a critical mass of users is desirable for attaining the full benefits of cooperative route planning. All entities are assumed to share a globally synchronized clock for correctly evaluating promise fulfillment and promise revocation requests. The required clock precision depends on the chosen timeslice granularity. For timeslices of several minutes and more, errors in the range of a few seconds are acceptable.

Concerning the number and deployment density of RAs (which is the main factor that the system operator can influence) there are two interesting extreme cases: the *dense deployment* case, where all major roads are split into multiple short segments and mapped to individual resources and the *sparse deployment* case, where RAs are only set up for common bottlenecks like tunnels, bridges or busy crossings. While the dense deployment case leads to a much higher granularity of promises and thus possibly allows for better optimizations, it is also tied to a larger initial investment for setting up RAs and a larger overhead for publishing and revoking promises per trip.

B. Security

1) Attacker model: In this section we investigate possible attacks on the functionality of our system. Our threat model is based on attackers that want to either disturb the system or exploit it for their own benefit (e.g., use it to redirect traffic). Due to these attacker goals, we assume that the organization maintaining the PB, PA and the RAs has no interest in colluding with such attackers. Specifically, we assume that an attacker can control only user entities. However, a strong attacker might be able to control multiple user entities, e.g., by registering under multiple fake or stolen identities.

2) Simple lying: A group of malicious users (respectively a strong attacker controlling multiple user identities) might want to influence traffic by publishing false promises. For example, they might collude to publish multiple identical promises concerning a specific resource, thus making it appear overburdened and causing other traffic participants to avoid it. The influence a group of malicious users can have on the global PB state is bounded by the number of PCs they have. It is expected that even if they can place a number of false promises before their PC pool is depleted, this will not influence routing decisions significantly. Users attempting such schemes will also run out of PCs quickly as they will not be able to redeem the promise tokens they received. They might request new PCs, but a non-compromised PA will blacklist them at some point, thus effectively banning them from the system and denying them the possibility to cause more mischief.

The potential threat stemming from large groups of user identities under malicious control is one of the main reasons for reissuing PCs only upon the verification of promise fulfilment. Simpler approaches, e.g., with PCs valid for only one day and automatically reissued afterwards, would allow such groups to disturb traffic on a continual basis.

3) Sybil Promises: In the *sybil promises* attack, an attacker publishes multiple identical promises, i.e., for the same resource and timeslice. In contrast to regular lying, he is honest about the promise and just cheating by anonymously publishing it multiple times, thus reserving more resources than he needs. The benefit for the attacker is that other traffic participants will perceive the resources along the attackers route as more crowded, thus potentially avoiding them and granting the attacker a road with less traffic. The effectiveness of sybil promises in comparison to regular lying depends on the possibility of realizing a sybil attack on the promise fulfillment mechanism. In other words, the possibility to trick a RA into mistaking a vehicle in its resource area for multiple vehicles. If this is feasible for a malicious user, he might be able to fulfill multiple identical promises simultaneously, effectively avoiding any loss of PCs in the process. Reliable techniques for identifying and counteracting sybil attacks in radio-based communication systems exist and have been evaluated for vehicular networking scenarios. See, for example, [11] for techniques based on radio-based position verification and [17] for an approach based on statistic signal strength distribution analysis. With such sybil protection mechanisms in place at RAs, the simultaneous fulfillment of multiple identical promises becomes infeasible.

Position-verification and similar countermeasures can also be used against wormhole attacks, where a user pretends to be at a given resource by using a proxy device physically placed

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2014.2361016, IEEE Internet of Things Journal

within the resource region. Specifically, if a radio source doesn't show the same mobility signature as other vehicles on the road, it can be blocked from attempting to fulfill promises.

4) Denial-Of-Service Attacks: A malicious group of users might try to attack the availability of the whole system by mounting a denial-of-service (DoS) attack against the PA or PB. The problem of preventing such attacks is comparable to preventing DoS attacks in many existing information and communication systems, e.g., popular websites or cloud-based navigation services. We therefore consider this challenge to be out of the scope of this paper. An attacker might also mount a DoS attack on a RA, e.g., by damaging it physically or by jamming its radio interface. As a consequence, users publishing promises concerning the resource managed by that RA will not be able to get a new PC upon promise fulfillment. Malfunctioning RAs can be blacklisted by the PA. For promises published before the RA was blacklisted, the PA can reimburse affected users by regenerating PCs for the blacklisted RAs without verifying the promises' fulfillment.

C. Privacy

1) Attacker model: For the analysis of the level of privacy offered by our approach, we will assume a strong adversary controlling the PA, the PB and all RAs. This corresponds to a service operator that is either malicious himself or is heavily colluding with a malicious entity. Users receive information only from the PB, so control over users is irrelevant in this context for an adversary controlling the PB.

The main goal of our adversary is to learn the locations of users - both past and planned future locations. One way is to analyze user requests at the PB and try to *determine their locations and plans based on PB queries*. Another is by intercepting the promise publication and promise fulfillment steps, i.e., trying to *link promises to user identities*. By doing this, he can learn both the past and the planned future whereabouts of individual users. If a direct linking between promises and user identities is not easily possible, the adversary might attempt such a linking by first combining several promises to trips and then using context knowledge to link such trips to individual users. For example, an adversary might correlate the location of a user's home with the start positions of trips he has observed. Thus, a secondary goal of the adversary is the *combination of multiple promises to trips*.

2) Query Privacy: The querying of the PB for information is a possible source of privacy loss for the requester. Information about his position and planned route might be deduced from his query. Multiple promising solutions to the problem of query privacy in location based systems have been discussed in the literature. See, e.g., [16] for a good overview on the topic. A simple solution is also to omit the querying step altogether and have the PB proactively broadcast aggregated information to users, similar to the Traffic Message Channel (TMC) approach widely used in navigation systems today.

3) Linking of promises to user identities: Given the use of a secure blind signature scheme for generating PCs and the fact that all promise-related user-actions are tied only to PCs and performed over anonymous communication channels, it is easy to see that a linking of promises to user identities is not easily possible. Specifically, real identities are only used for getting initial batches of blindly signed PCs. For honest users, this step will need to be performed very rarely and the only information gained from it by the adversary is that the user is participating in the system at all. PCs generated upon promise fulfillment are also signed blindly by the PA, so PCs by the same user cannot be linked together.

4) Combination of multiple promises to trips: If an adversary can link the promises by one user and thus reconstruct his intended route, he might be able to link that route back to the user using additional context knowledge. Since PCs are not linkable between each other, the linking can only happen by correlating the promises themselves, either upon publication to the PA or upon fulfillment at RAs.

All promises by a user are communicated to the PA using different anonymous channels so that the linking of promises based on communication metadata is not easily possible. However, timing-based correlation attempts might be possible, i.e., by grouping together promises arriving in close succession. As a simple protection mechanism, navigation devices can introduce jitter between promise publications and decouple the order in which promises are published from the order in which they will be fulfilled.

As an alternative to observing the promise publication step, the adversary might collect information from multiple RAs and attempt to link promises based on the time of their fulfillment. This is mostly equivalent to using radio-enabled RSUs to log the passing of vehicles. There is a plethora of work dealing with privacy in this classical vehicular networking scenario. In [10], for example, Freudiger et al. propose the use of pseudonyms per vehicle and the changing of pseudonyms whenever vehicles pass through predefined mix zones. In our proposal, communication happens only at RAs and the promises being used for authentication can be viewed as single-use pseudonyms that are implicitly changed between exchanges with different RAs. Thus, our approach maintains at least the same level of privacy in this aspect as state of the art vehicular-networking approaches.

The difficulty and impact of linking promises together is, for both attack methods, largely dependent on the specific practical deployment conditions of our proposed system. Given a sparse deployment of RAs and a large user population, we expect the reconstructability of routes and the linkability of routes to user identities to be negligible.

D. Conclusion

Our analysis confirms that following non-trivial security and privacy properties are simultaneously guaranteed by our scheme: (1) Assuming the honesty of the service operator and the integrity of the entities under his control, malicious users cannot effectively abuse the cooperative route planning system for altering traffic flow. (2) Individual users participate in the system in an anonymous fashion, with no user actions or published routing information being linkable to them. (3) Individual routes, which can potentially be linked back to user identities, are not easily reconstructible by the service operator.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2014.2361016, IEEE Internet of Things Journal

V. Performance Evaluation

We identify two main potential performance bottlenecks that are specific to our approach: the *computation overhead* introduced by the extensive use of cryptographic primitives and the additional latency introduced by the use of different *anonymous secure communication channels* for each user operation. We do not evaluate the overhead of mechanisms common to other systems (e.g., the distribution of traffic state via the PB), as we believe them to be well understood and either irreplaceable for realizing any cooperative route planning system or negligible in terms of their impact on performance. Similarly, we also restrain from evaluating the communication overhead of our scheme. The small (easily below 1 kilobyte), infrequent messages used in our approach are unlikely to cause either an overburdening of the short-range radio link to RAs or significant costs when communicating over cellular network links with the PA.

A. Computation Overhead

For evaluating the computation overhead of our approach, we implemented key steps from the mechanisms described in Sec. III-C that rely heavily on the use on cryptographic primitives. Specifically, we implemented the following operations: generating and blinding a coin ID (user), signing a blinded coin ID (PA), unblinding a blinded coin ID signature (user), validating a coin ID signature (user/PA), signing a blinded coin ID and generating a promise token with it (PA), validating a promise token signature (user/RA/PB) and cashing in a promise token, decrypting the coin ID signature contained in it (RA). Our implementation is based on RSA and RSA blind signatures as described in the Appendix. We evaluated key sizes of 1024 as well as 2048 bit. All user operations were evaluated on a low-end Android smartphone (ARM-based CPU at 600 MHz) to model the use of a device with low computational resources. The remaining operations were evaluated on a regular desktop computer (AMD Athlon II X4 CPU at 3.01 GHZ per core, the evaluation was single-threaded). All operations were repeated 100 times with different cryptographic keys and their median execution time was logged. We do not evaluate the power consumption of our protocol at the user side, as we expect the user side operations to be performed inside a vehicle, in which case the power usage of any smartphone-class device is negligible.

The results of our measurements can be found in Tab. II. It can be seen that for 1024-bit RSA, all operations except the generation of a promise token require less than 10 ms and for 2048-bit, less than 50 ms to complete on the respective device class they are likely to run on. The generation of a promise token involves the computation of two cryptographic signatures - one for the new PC and one for the promise token base itself. Hence, and because in the RSA cryptosystem signing is more expensive than signature verification, the median computation time here reaches 12.87 ms for 1024-bit RSA and 88.9 ms for 2048-bit RSA. Even for 2048-bit RSA, a PA will still be able to easily process more than 10 promise requests per second and CPU core. For perspective, given a population of 1 million simultaneously active users, an average deployment density of one RA per 10 km and an average vehicular movement speed of 100 km/h, an average of about 280 promises are likely to arrive at the PA per second. Thus, with our implementation and key lengths of 2048 bit, computational resources equivalent to 28 CPU cores (one high-end server) will need to be deployed by the service operator. This requirement can be reduced by several orders of magnitude by employing specialized cryptographic hardware. Also, more efficient (but slightly more complex) blind signature schemes exist, e.g., [13].

TABLE II
Computation times with implementation based on RSA.

Operation	Time (ms)	
	1024-bit RSA	2048-bit RSA
signing blinded coinID (PA)	6.28	43.62
validating promise coin (PA)	0.61	1.11
generating promise token (PA)	12.87	88.9
validating promise token (RA/PB)	0.3	1.55
cashing in promise token (RA/PA)	6.55	43.73
generating blinded coinID (user)	2.15	5.59
unblinding coinID signature (user)	6.22	23.13
validating promise coin (user)	1.27	4.56
validating promise token (user)	1.38	4.58

B. Anonymous Secure Communication Channel Latency

The unlinkability of user actions to communication meta-data like IP addresses is an important element of the privacy level offered by our approach. Additionally, communication channels should be secure from end to end, to prevent attackers from disturbing the functionality of the system. These requirements can potentially lead to a significant increase in communication latency, both for setting up a communication channel and for sending data over it. For deriving an estimate for the scale of these latencies, we evaluated the approximate communication latency of user-PA interactions via the anonymous secure communication channel described in Sec. III-C1. We constructed an evaluation scenario to measure the time required for establishing an anonymous secure communication channel over a cellular network link and transferring 1 kilobyte of data over this channel. The 1 kilobyte data transfer was chosen for deriving an estimate for the latency involved in a common user-PA operation like the publishing of a promise. We used a HTTPS (HTTP on top of TLS/SSL) download of a 1 kilobyte file to model the latency of setting up a secure TLS/SSL connection and transferring 1 kilobyte of data over it. We measured the time for performing this file transfer without Tor, the time for bootstrapping a Tor circuit and the compound time for bootstrapping a Tor circuit and performing the file transfer over the Tor connection. We performed our measurements on 7 different days during different times of day. In total, each measurement was repeated 70 times (10 measurements on each measurement day). All measurements were performed using a cellular network interface using the EDGE standard.

The results of our measurements are presented in Fig. 2. The figure depicts the median of all measured times together with the respective minimum and maximum times measured during our evaluation. According to the measurements, establishing

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2014.2361016, IEEE Internet of Things Journal

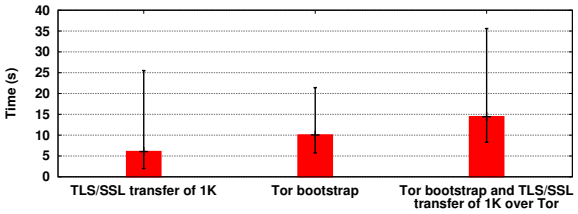

Fig. 2. Secure and anonymous communication channel latency.

an anonymous and secure channel using a EDGE-based cellular network link and performing an operation like a promise publication over it takes between 10 and 37 seconds, with a median duration of 15 seconds. Compared to the baseline latency in our evaluation scenario - the data transfer over a secure but not anonymous communication channel (2-26 seconds, median of 6 seconds) - this is a median increase of only about 9 seconds. In any case, all measured latency values are, by a large margin, acceptable for a cooperative route planning application. Route planning is typically done well in advance and operates on a timescale of tens of minutes instead of on a timescale of seconds.

VI. Related Work

Cooperative, anticipatory route planning has been widely discussed in the context of intelligent transportation systems [6], [9]. Existing works focus on improving route planning decisions based on collected planning data from users. Proposed systems are evaluated using extensive traffic simulations. Evaluation results demonstrate that significant improvements in average travel times are possible, especially in scenarios with a high traffic density. While these contributions are highly valuable for advancing the cooperative route planning concept, no works exist that consider the issue of protecting user privacy while minimizing the systems' vulnerability to abuse.

There is a plethora of works on location privacy and privacy for collaborative sensing that focus on protecting privacy while performing operations on a user's current location and state. See, for example, [16] for a good overview of location privacy approaches. While there are many works dealing with privacy in systems reporting a user's current state, very few works are known to the authors that deal with intention privacy. According to our knowledge, we are the first to tackle the specific problem of publishing planned routes in a vehicular route planning context while protecting the users' privacy and preventing malicious behavior.

In [4], Chim et al. propose a scheme for making power usage reservations in a smart grid context using blindly signed anonymous credentials. While their scenario and approach are similar to ours, their scheme has important drawbacks. For example, it only guarantees that users will consume the reserved amount of energy per day and not whether they will consume it at the specific timeslices they claim. Additionally, the electricity provider in [4] is assumed to know the total daily power consumption of a user. In a traffic scenario, this would correspond to the traffic authority or a private company knowing the distance traveled by each participating vehicle, which is undesirable from a privacy standpoint and not easily realizable. The idea of preventing abuse using single-spend, renewable tokens has also appeared in the context of privacy-preserving subscription services (e.g., [15]). However, the question of realizing anonymous promises and reservations has not been raised in these works.

Privacy-preserving reputation systems have been proposed for preventing abuse in systems with pseudonymous participants. Androulaki et al., for example, propose an approach centered around the idea of *repcoins* [1], which are also based on Chaum electronic cash. However, their scheme does not consider the promise scenario, i.e., is lacking a mechanism for publishing promises in such a way that reputation can be gained on promise fulfillment without enabling an adversary to link the promise back to its author. The *IncogniSense* framework [5] enables reputation transfers between pseudonymous identities that preserve the unlinkability between pseudonyms. With this, the same pseudonym could be used for publishing and fulfilling individual promises, with the pseudonym's reputation score incrementing upon fulfillment. In order to avoid linkability between promises however, a different pseudonym will have to be used per promise. This is difficult to reconcile with IncogniSense's periodic pseudonym switching mechanism and significantly more complex than our approach.

VII. Conclusion and Future Work

Realizing advanced services while maintaining user privacy is a central tension field in IoV research. Here, we proposed a scheme for sharing planned routes in a cooperative route planning context that offers both strong privacy guarantees and security against abuse. According to our knowledge, we are the first to consider the problem of privacy and abuse-prevention in cooperative route planning systems where participants publish information about their planned routes. In our approach, plans are published anonymously as a series of promises concerning segments of the planned route. Security is realized by requiring the use of blindly signed *promise coins* for making each promise. New promise coins are issued upon promise fulfillment. Thus, honest users retain their right to participate in the system while malicious users get banned quickly. Promise coins are always signed blindly and thus not linkable to the user they have been issued to. Consequently, promises are also unlinkable to users and between each other. We provided a detailed description of our proposal and a thorough security and privacy analysis. Through performance measurements of the involved cryptographic operations and latency measurements of the involved anonymous communication processes, we show that privacy-preserving cooperative route-planning is practically feasible.

References

[1] E. Androulaki, S. G. Choi, S. M. Bellovin, and T. Malkin, "Reputation systems for anonymous networks," in *Privacy Enhancing Technologies*. Springer, 2008, pp. 202–218.

[2] D. Chaum, "Blind signatures for untraceable payments." in *Crypto*, vol. 82, 1982, pp. 199–203.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2014.2361016, IEEE Internet of Things Journal

[3] D. Chaum, "Security without identification: Transaction systems to make big brother obsolete," *Communications of the ACM*, vol. 28, no. 10, pp. 1030–1044, 1985.
[4] T. W. Chim, S.-M. Yiu, L. C. Hui, and V. O. Li, "Privacy-preserving advance power reservation," *Communications Magazine, IEEE*, vol. 50, no. 8, pp. 18–23, 2012.
[5] D. Christin, C. Roßkopf, M. Hollick, L. A. Martucci, and S. S. Kanhere, "Incognisense: An anonymity-preserving reputation framework for participatory sensing applications," *Pervasive and Mobile Computing*, 2013.
[6] R. Claes, T. Holvoet, and D. Weyns, "A decentralized approach for anticipatory vehicle routing using delegate multiagent systems," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 12, no. 2, pp. 364–373, 2011.
[7] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2," RFC 5246 (Proposed Standard), Internet Engineering Task Force, 2008.
[8] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *13th USENIX Security Symposium*. Usenix, 2004.
[9] K. Dresner and P. Stone, "Multiagent traffic management: An improved intersection control mechanism," in *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*. ACM, 2005, pp. 471–477.
[10] J. Freudiger, M. Raya, M. Felegyhazi, P. Papadimitratos, and J. P. Hubaux, "Mix-zones for location privacy in vehicular networks," in *Proceedings of the First International Workshop on Wireless Networking for Intelligent Transportation Systems (Win-ITS)*, 2007.
[11] P. Golle, D. Greene, and J. Staddon, "Detecting and correcting malicious data in vanets," in *Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*. ACM, 2004, pp. 29–37.
[12] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, and T. Weil, "Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions," *Communications Surveys & Tutorials, IEEE*, vol. 13, no. 4, pp. 584–616, 2011.
[13] T. Okamoto, "Efficient blind and partially blind signatures without random oracles," in *Theory of Cryptography*. Springer, 2006, pp. 80–99.
[14] J. Rybicki, B. Scheuermann, M. Koegel, and M. Mauve, "Peertis: a peer-to-peer traffic information system," in *Proceedings of the sixth ACM international workshop on VehiculAr InterNETworking*. ACM, 2009, pp. 23–32.
[15] S. G. Stubblebine, P. F. Syverson, and D. M. Goldschlag, "Unlinkable serial transactions: protocols and applications," *ACM Transactions on Information and System Security (TISSEC)*, vol. 2, no. 4, pp. 354–389, 1999.
[16] M. Wernke, P. Skvortsov, F. Dürr, and K. Rothermel, "A classification of location privacy attacks and approaches," *Personal and Ubiquitous Computing*, pp. 1–13, 2012.
[17] B. Xiao, B. Yu, and C. Gao, "Detection and localization of sybil nodes in vanets," in *Proceedings of the 2006 workshop on Dependability issues in wireless ad hoc networks and sensor networks*. ACM, 2006, pp. 1–8.

APPENDIX

In the following, we give an example realization of some of the cryptographic functions used in Section III-C. Specifically, we define the functions required for the blind signing of promise coins, namely $\mathrm{blind}()$, $\mathrm{sig}_{\mathrm{CIC}}()$ and $\mathrm{unblind}()$. As in the remainder of the paper, we use RSA-based signatures here, as well as RSA blind signatures as introduced in [3].

The primitives $blind()$ and $unblind()$ are the cornerstone of the blind signature concept. Together with the coin signing function $sig_{CIC}()$ (that generates signatures with the *Coin Issuing Certificate* (CIC)), they satisfy the following equation (for an arbitrary message m):

$$\mathrm{unblind}(\mathrm{sig}_{\mathrm{CIC}}(\mathrm{blind}(m))) = \mathrm{sig}_{\mathrm{CIC}}(m)$$

In the following, we assume that the employed CIC is based on an RSA key with d and e being the private and public parts of this key and N being its public modulus. With a random r relatively prime to N, the blinding of m can then be realized as:

$$\mathrm{blind}(m) := mr^e \pmod N$$

In order for the unblinding step to work, $sig_{CIC}()$ must be implemented without any modifications (e.g., padding or hashing) on the message m after it has been transmitted to the signer (in our case, the Promise Authority). Thus, we use the plain RSA signature scheme for $sig_{CIC}()$:

$$\mathrm{sig}_{\mathrm{CIC}}(m) := m^d \pmod N$$

$$\mathrm{sig}_{\mathrm{CIC}}(\mathrm{blind}(m)) := (\mathrm{blind}(m))^d = m^d r^{ed} \pmod N$$

For $\mathrm{unblind}()$ we apply the inverse of r to the signature, thus receiving a valid signature on m:

$$\mathrm{unblind}(\mathrm{sig}_{\mathrm{CIC}}(\mathrm{blind}(m))) := m^d r^{ed} r^{-1} = m^d \pmod N$$

There are known dangers to using blind RSA signatures. If the key used for blind signing is used for encryption, an attacker can trick the signer to decrypt arbitrary bits of encrypted data. Thus, we use a dedicated key for signing promise coins - the CIC. A second danger lies in the commutativity of unpadded RSA signatures as required for the RSA blind signature scheme. It can be exploited to generate more than one valid message-signature pair during the blind signing process. For example, a user can easily use multiple blinding factors on the same message. Unblinding separately with each blinding factor then yields multiple distinct messages with valid signatures. Thus, we recommend the use of the hash and sign paradigm in a real implementation: blinding and signing not the promise coin itself, but a cryptographic hash of it.