Privacy-Preserving Blockchain-Based Energy Trading Schemes for Electric Vehicles

Mohamed Baza, Ahmed Sherif, Mohamed M. E. A. Mahmoud, Spiridon Bakiras,
Waleed Alasmary, *Senior Member, IEEE*, Mohamed Abdallah, and Xiaodong Lin, *Fellow, IEEE*

***Abstract*—An energy trading system is essential for the successful integration of Electric Vehicles (EVs) into the smart grid. In this paper, leveraging blockchain technology, we first propose a privacy-preserving charging-station-to-vehicle (CS2V) energy trading scheme. The CS2V scheme is useful in crowded cities where there is a need for a charging infrastructure that can charge many EVs daily. We also propose a privacy-preserving vehicle-to-vehicle (V2V) energy trading scheme. The V2V scheme is useful when charging stations are not available or far and cheaper prices can be offered from EVs, e.g., if they charge from renewable energy sources. In the V2V scheme, the privacy of both charging and discharging EVs including location, time, and amount of power are preserved. To preserve privacy in both schemes, EVs are anonymous, however, a malicious EV may abuse the anonymity to launch Sybil attacks by pretending as multiple non-exiting EVs to launch powerful attacks such as Denial of Service (DoS) by submitting multiple reservations/offers without committing to them, to prevent other EVs from charging and make the trading system unreliable. To thwart the Sybil attacks, we use a common prefix linkable anonymous authentication scheme, so that if an EV submits multiple reservations/offers at the same timeslot, the blockchain can identify such submissions. To further protect the privacy of EV drivers, we introduce an anonymous and efficient blockchain-based payment system that cannot link individual drivers to specific charging locations. Our experimental results indicate that our schemes are secure and privacy-preserving with low communication and computation overheads.**

***Index Terms*—Security, privacy, blockchains and smart contracts, energy trading, and electric vehicles (EVs).**

I. Introduction

THE smart grid is a revolutionary upgrade to the traditional electricity grid that aims to create a clean, resilient, and efficient system by integrating information technology, data communication, sensing, and control technologies into the power system [1]. The smart grid promotes high penetration level of renewable energy sources and EVs to reduce greenhouse gas emissions. Specifically, residents can install solar panels on their rooftops to generate electricity to power their homes and charge their EVs. EVs are becoming increasingly popular and it is expected that most of the vehicles will be electric in the future [2].

EVs can charge at home from electrical grid and solar panels [3]. Another charging approach is to charge from charging stations (CSs). This form of charging is called charging stations to vehicle (CS2V) energy trading, where energy traders can offer competitive prices to the EVs [4]. CS2V is useful for providing fast charging or when an EV is travelling for long distance. Alternatively, EVs with surplus energy can charge other EVs, and this form of charging is called vehicle-to-vehicle (V2V) energy trading [5]. This approach is useful when charging stations are far or unavailable, e.g., in developing countries or remote areas, and do not have sufficient energy resources, or when cheap prices can be offered by the EVs because they can charge from cheap renewable energy sources.

To facilitate charging of EVs, an energy trading system is needed to match the bids of energy sellers to the requests of buyers and connect them. In these systems, a central system (server) that acts as service manager is usually used. To organize the energy trading, both the energy sellers and buyers need to exchange sensitive information with the service manager about the location and time of energy trading. Without privacy protection, the untrustworthy server and malicious adversaries can infer sensitive information about the EVs' drivers. For example, if the charging stations are installed at medical clinic parks, working place, and hospitals, sensitive information about the EVs' drivers can be revealed such as their habits, workplaces, and health conditions. Moreover, the server can infer if the users are on travel by monitoring their trading activities, e.g., if they do not buy or sell energy for a long time. Furthermore, it has been shown that the server can infer the real identity

Manuscript received August 15, 2020; revised December 6, 2020 and January 30, 2021; accepted April 4, 2021. Date of publication July 20, 2021; date of current version September 17, 2021. This work was supported in part by the Deanship of Scientific at Umm Al-Qura University under Grant 19-COM-1-01-0016, NSF under Grants 1619250, 1618549, and 1852126 and NSF grants 1619250, 1618549, and 1852126, and part of this paper, specifically Sections I, III, IV, V, VII supported by NPRP under Grant NPRP12S-0221-190127 from the Qatar National Research Fund (a member of Qatar Foundation). The review of this article was coordinated by Prof. Zibin Zheng. *(Corresponding author: Ahmed Sherif.)*

Mohamed Baza is with the Department of computer science, College of Charleston, Charleston, SC 29424 USA (e-mail: mohbaza@hotmail.com).

Ahmed Sherif is with the School of Computing Sciences and Computer Engineering, University of Southern Mississippi, Hattiesburg, MS 39402 USA (e-mail: ahmed.sherif@usm.edu).

Mohamed M. E. A. Mahmoud is with the Department of Electrical and Computer Engineering, Tennessee Tech University, Cookeville, TN 38505 USA (e-mail: mmahmoud@tntech.edu).

Spiridon Bakiras and Mohamed Abdallah are with the Division of Information and Computing Technology, College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar (e-mail: sbakiras@hbku.edu.qa; moabdallah@hbku.edu.qa).

Waleed Alasmary is with the Department of Computer Engineering, Umm Al-Qura University, Makkah 21421, Saudi Arabia (e-mail: wsasmary@uqu.edu.sa).

Xiaodong Lin is with the School of Computer Science, University of Guelph, Guelph, Ontario N1G 1Y4, Canada (e-mail: xlin08@uoguelph.ca).

Digital Object Identifier 10.1109/TVT.2021.3098188

of EVs' drivers using the locations visited by them [6], and therefore, using anonymization alone is not enough to preserve privacy.

In addition, running the service by a central server makes the system vulnerable to the single point of failure and several cyber attacks [7], [8]. For instance, Uber has witnessed a tremendous data leakage of 57 million customers and drivers for more than a year. The company has paid 148 million to settle an investigation on the data breach [9]. Also, the centralized platform suffers from a lack of transparency since the server can favor certain buyers to be served first and certain sellers to sell their energy first. Furthermore, if the security of the service manager is compromised, the service can be interrupted and the users' data can be disclosed, altered, or even deleted. In addition, relying on existing payment systems (such as credit or debit cards) for energy trading transactions may violate the privacy of the EVs' drivers, because their charging locations can be known or tracked over long periods of time. On the other hand, utilizing cryptocurrencies such as Bitcoin and Zcash [10] is not a viable solution, since they are prohibited or restricted in some countries [11]. Moreover, such blockchain platforms implement consensus protocols that demand high computational power and they do not provide efficient processing of the transactions.

In contrast to the traditional client-server model, Blockchain is a verifiable, immutable and distributed ledger that allows mistrusting entities to transact with each other without relying on a central third party and with no single point of failure problem. Blockchain is a transparent data structure that is organized as a chain of blocks and managed by a network of computers, called miners, running a peer-to-peer (P2P) protocol. Each block contains a set of transactions that are committed by the network peers according to a predefined consensus algorithm [12]. Instead of using a central architecture for the energy trading system, there are some works that use blockchain to design energy trading systems [5], [13]–[15] but they mainly focus on optimization techniques to maximize the sellers' profit. Very few works, such as [4], [13], have studied the security and privacy issues in the energy trading. The proposed scheme in [13] consider only security and it does not consider privacy and charging reservations. The scheme proposed in [4] suffers from several problems and limitations. First of all, the selected charging stations do not know if they are selected since the reservation is made hidden on the blockchain and this may lead to scheduling conflict since two different EVs may select the same bid. In addition, the paper does not consider the V2V energy trading case where the privacy of both charging/discharging EVs should be protected. Finally, the paper does not develop an anonymous payment method to enable payment while protecting the privacy of the EVs' drivers.

To tackle the above challenges, in this paper, by leveraging blockchain and smart contract technology, we first propose a privacy-preserving CS2V scheme to enable energy trading between CSs and EVs. Then, we propose a privacy-preserving V2V energy trading scheme to enable EVs (called discharging EVs) to charge other EVs (called charging EVs). In the CS2V scheme, the CSs publish energy bids and EVs select an appropriate bid and reserve it. To improve the efficiency and scalability of the scheme, we choose a *private* blockchain made by the charging stations to run the scheme. In the privacy-preserving V2V energy trading scheme, we use a public blockchain which is appropriate for the V2V setting because an infrastructure to form a private blockchain does not exist. In this scheme, a charging vehicle sends a charging request with the region of interest and the time to find trading offers to the blockchain. Then, interested discharging vehicles send encrypted bids including the amount of power to sell and price to the blockchain. A charging EV, then, selects the best bid(s) and sends a reservation request to the blockchain.

Moreover, although anonymity is important to preserve the privacy of the EVs' drivers, some EVs may abuse this anonymity to launch Sybil attacks to pretend as multiple non-existing EVs and then launch powerful attacks on the system such as Denial of Service (DoS) attack. Specifically, the Sybil EVs may launch a DoS attack by making many fake reservations to block other EVs from charging and thus make the energy trading system unavailable. To thwart the Sybil attacks, we leverage a common prefix linkable anonymous authentication [16], so that a vehicle is allowed to authenticate its messages anonymously, however, if it tries to pretend as multiple EVs and submit multiple messages (reservations) with the same prefix, i.e., timeslot, the blockchain can link these submissions and know that they are sent from the same EV. However, no one can link the messages that are sent from the same vehicle at different times to preserve privacy.

In addition, we develop an anonymous and efficient blockchain-based payment system that is based on real currency and integrate it in our scheme. In particular, the system leverages a financial institution (FI) to exchange real currency with digital coins that are provably untraceable. During purchasing the coins, FI can know the real identity of the buyer but when the coins are deposited by a charging station, FI can not link coins to the buyers to preserve their location privacy. Also, coin ownership can be transferred to enable EVs to pay for charging without involving FI, and thus our system does not require involving the FI in the payment of each transaction for efficiency and scalability. In addition, our system is secure against unauthorized use of coins, by enforcing a proof of ownership for a batch of coins using a zero knowledge proof (ZKP) protocol instead of verifying ownership of individual coins for efficiency. Moreover, our system is secure against double spending using blockchain which stores the hash of spent coins.

Our main contributions and the challenges the paper aims to address can be summarized as follows.

- We propose blockchain-based energy trading schemes for CS2V and V2V that preserve the privacy of EVs' drivers. We also secure our schemes against Sybil attacks that can be used to launch powerful attacks, such as DoS, that threats the service availability.
- An anonymous and efficient blockchain-based payment system is developed and integrated into our schemes so that the EVs drivers' privacy is preserved and double-spending and stolen coin spending attacks are thwarted. The payment is also efficient since the ownership of batch of coins can be verified at once through ZKP protocol and FI does not need to be involved in every transaction.

- Extensive simulations and analysis are conducted to evaluate the proposed schemes. The results demonstrate that our schemes are secure and preserve EVs drivers' privacy with acceptable communication/computation overheads.

A preliminary version of this paper has been published in [17]. The main differences between [17] and this paper are as follows. Firstly, this paper addresses the energy trading among EVs and proposes a privacy-preserving energy trading V2V scheme, but [17] considers only the case of CS2V charging. Secondly, EVs can be anonymously authenticated and this paper provides a countermeasure to Sybil attack. Thirdly, we changed the payment system, so that the FI does not need to be involved in every transaction. Finally, more analysis and simulation results have been added in this paper.

The rest of this paper is organized as follows. In Section II, we discuss some techniques that are used in our schemes. Section III discusses the considered network and threat models and design goals. Then, our proposed schemes are presented in details in Section IV. In Section V, we present the security and privacy analysis of our schemes followed by performance evaluations. Section VI discusses the related works. Finally, we give concluding remarks in Section VII.

II. Preliminaries

In this section, we present the necessary background needed to understand this paper.

A. Blind Elliptic Curve DSA Signatures

Using a blind signature cryptosystem, the user (*requester*) can obtain a valid signature on a message M from the *signer* without revealing M to the signer. In our payment system, we leverage the blind signature cryptosystem described in [18], which is based on an elliptic curve implementation of the digital signature algorithm (DSA) to create anonymous coins because it offers faster computations with significantly shorter signatures. The aforementioned cryptosystem consists of the following steps.

1) All parties use the same elliptic curve of order n with generator G.[1] In addition, the signer's public key is $P = d{\cdot}G$, where d is the corresponding private key and $d \in \mathbb{Z}_n^*$.
2) The signer selects a uniformly random element $k \in \mathbb{Z}_n^*$ and sends $R = k{\cdot}G$ to the requester.
3) The requester selects uniformly random elements $\gamma, \delta \in \mathbb{Z}_n^*$ and computes $A = R + \gamma{\cdot}G + \delta{\cdot}P$. Let x be the x-coordinate of point A, and $t = x \bmod n$. The requester computes $c = H(M||t) \bmod n$ and sends $c' = (c - \delta) \bmod n$ to the signer. $H(\cdot)$ is a cryptographically secure hash function, and $H : \{\}^* \rightarrow Z_n^*$.
4) The signer computes $s' = (k - c'{\cdot}\mathrm{d}) \bmod n$ and sends the result back to the requester.
5) The requester computes $s = (s' + \gamma) \bmod n$ and stores the signature of M as (s, c).
6) To verify the signature, the *verifier* computes $A = c{\cdot}P + s{\cdot}G$. Then, it computes $t = x \bmod n$, where x is the x-coordinate of point A. The verifier checks if $c \overset{?}{=} H(M||t) \bmod n$.

B. Schnorr's Identification Protocol

An identification protocol allows the owner of a public key (*prover*) to prove to a *verifier*, in zero knowledge, that he indeed knows the value of the underlying secret key. A well-known identification protocol is called Schnorr [19], which is summarized below for the case of an elliptic curve cryptosystem.

1) We assume that all parties use the same elliptic curve of order n with generator G. The prover's public key is $P = d{\cdot}G$, where d is the corresponding private key.
2) The prover selects a uniformly random element $k \in \mathbb{Z}_n^*$ and sends $R = k{\cdot}G$ to the verifier (*commitment*).
3) The verifier selects a random element $e \in \mathbb{Z}_n^*$ and sends it to the prover (*challenge*).
4) The prover computes $s = (k + e{\cdot}\mathrm{d}) \bmod n$ and sends it to the verifier (*response*). The verifier accepts the proof if $s{\cdot}G = R + e{\cdot}P$.

Gennaro *et al.* [20] use higher degree polynomials that enable the execution of multiple Schnorr protocol instances simultaneously with overhead that is very close to the overhead of a single instance. The protocol is identical to the one described above, except for the last step. In particular, the prover holds m public keys $\{P_1, P_2, \ldots, P_m\}$, corresponding to private keys $\{d_1, d_2, \ldots, d_m\}$. When the prover receives the verifier's challenge (e), it computes the response $s = (k + \sum_{i=1}^{m} e^i \cdot d_i) \bmod n$. The verifier then accepts the proof by checking if $s \cdot G \overset{?}{=} R + \sum_{i=1}^{m} e^i \cdot P_i$.

In our schemes, we uses the batch version of Schnorr's protocol to allow the owner of coins, i.e., EV/CS (prover) to deposit the digital coins into their account efficiently by proving the ownership of the coins to the FI (verifier). Also, the scheme can be extended to allow the verifier to verify the proofs of different provers at once by simply checking if $(s_1 + s_2) \cdot G \overset{?}{=} R_1 + R_2 + \sum_{i=1}^{m} e^i \cdot P_i + \sum_{j=1}^{m} e^j \cdot P_j$, where (R_1, s_1) and (R_2, s_2) are the commitment and response values from two provers, respectively.

C. Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARK)

A zero-knowledge proof (zk-proof) allows a party (prover) to generate a cryptographic proof convincing another party (verifier) that some values are obtained by faithfully executing a pre-defined operations on some private inputs (witness) without revealing any information about the witness [21]. The zk-SNARK further allows such a proof to be generated non-interactively. More importantly, the proof is *succinct*, i.e., the proof size is independent on the complexity of the statement to be proved. More precisely, zk-SNARK has three phases. The setup phase outputs the public parameters to establish a SNARK for a NP-complete language L. In the proof generation phase,

[1] Throughout the paper, we use uppercase letters to represent elliptic curve points, and lowercase letters to represent scalars.

the $Prover$ uses an instance x, and witness w to generate a proof for the statement $x \in L$. Finally, in the verification phase, the $Verifier$ can efficiently verify the proof.

D. Common-Prefix-Linkable Anonymous Authentication

Recenlty, Lu et. al, [16] have proposed a common prefix linkable anonymous authentication scheme based on zk-SNARK descried in Section II-C. In this scheme, a user can authenticate messages anonymously and prove the validity of his certificate without being identified and without linking the messages that are sent from the same user. The only exception is when the same user authenticates messages with the same prefix. In this case, it can be known that messages are sent from the same user. The scheme consists of the following five algorithms:

- $Setup(1^{\lambda}) \rightarrow (PP, msk, mpk)$: This algorithm establishes the public parameters PP needed for the zk-SNARK scheme and the system's master public key mpk and master secret key msk.
- $CertGen(msk, PK_i) \rightarrow cert_i$: This algorithm outputs certificate $cert_i$ to validate the public key PK_i of a private key SK_i.
- $Auth(p||m, SK_i, PK_i, cert_i, PP) \rightarrow \pi = (t_1, t_2, \eta)$: This algorithm generates an attestation π on a message m and a prefix p that the sender indeed learns a secret key corresponding to a valid certificate $cert_i$ without being able to identify the sender. The algorithm first computes two tags, $t_1 = H_1(p, SK_i)$ and $t_2 = H_1(p||m, SK_i)$, where H_1 is a secure hash function, e.g., SHA-256. Then, let $\vec{w} = (SK_i, PK_i, cert_i)$ represents the private witness, and $\vec{x} = (p||m, mpk)$ be all common knowledge, the algorithm runs zk-SNARK proving algorithm $Prover(\vec{x}, \vec{w}, PP)$ for the following language $\mathcal{L}_T = \{t_1, t_2, \vec{x} = (p||m, mpk) \mid \exists \vec{w} = (SK_i, PK_i, cert_i)$ s.t. $CertVrfy(cert_i, PK_i, mpk) = 1 \wedge pair(PK_i, SK_i) = 1 \wedge t_1 = H_1(p, SK_i) \wedge t_2 = H_1(p||m, SK_i)\}$, where the $CertVrfy$ algorithm checks the validity of the certificate using a signature verification, and $pair$ algorithm verifies whether two keys are a public/secret key pair. Finally, the algorithm outputs $\pi = (t_1, t_2, \eta)$.
- $Verify(p||m, \pi, mpk, PP) \rightarrow \{0, 1\}$: this algorithm runs the zk-SNARK $Verifier$ algorithm on π and PP, and outputs $0/1$ to indicate whether the attestation is valid or not for the attested message.
- $Link(m_1, \pi_1, m_2, \pi_2) \rightarrow \{0, 1\}$: On inputting two attestations $\pi_1 = (t_1^1, t_2^1, \eta_1)$ and $\pi_2 = (t_1^2, t_2^2, \eta_2)$, the algorithm simply checks $t_1^1 \overset{?}{=} t_1^2$. If the check is true, the algorithm outputs 1, indicating that π_1 and π_2 are computed from the same user on the same prefix; otherwise, it outputs 0, indicating that π_1 and π_2 are computed by different users. We also use $Link(\pi_1, \pi_2)$ for short. This operation is very efficient because it does not need cryptographic operations.

III. Network/Threat Models and Design Goals

In this section, we present the considered network model followed by the adversary and threat models, and then, we introduce the design goals of our schemes.

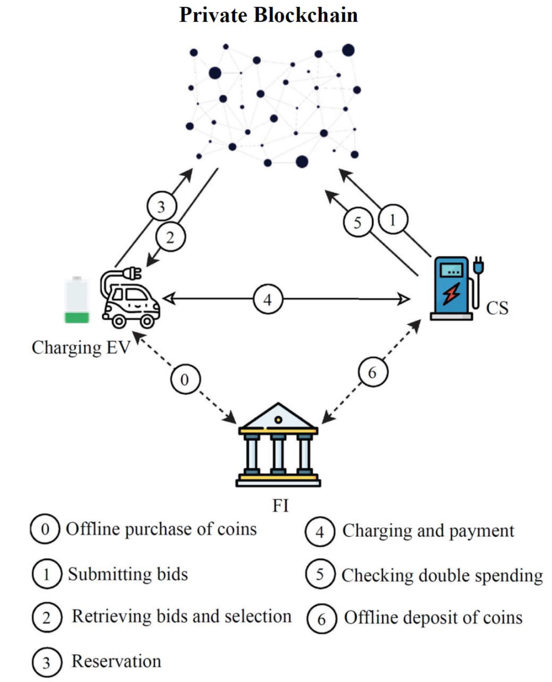

Private Blockchain
3
2
5
1
4
Charging EV
CS
0
6
FI
0 Offline purchase of coins
1 Submitting bids
2 Retrieving bids and selection
3 Reservation
4 Charging and payment
5 Checking double spending
6 Offline deposit of coins

Fig. 1. Network model of the CS2V scheme.

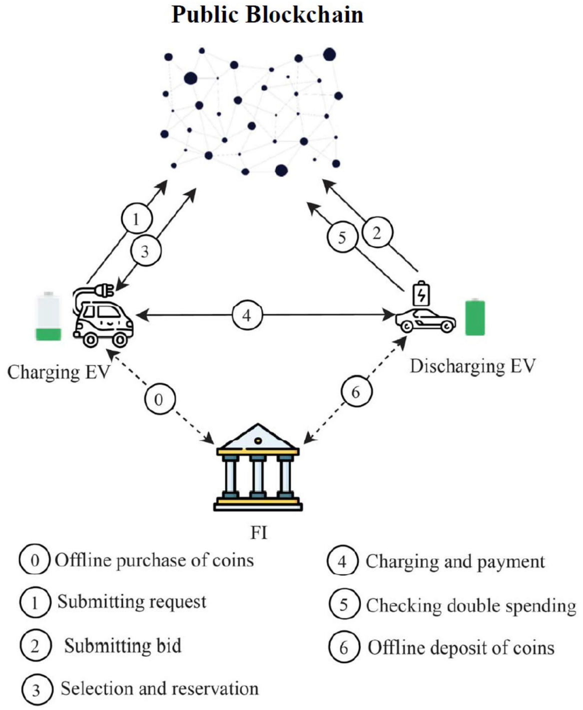


Fig. 2. Network model of the V2V scheme.

A. Network Models

Figs. 1 and 2 illustrate the network models of the CS2V and V2V schemes, respectively. The main entities are charging stations, charging EVs, discharging EVs and the financial institution (FI). In addition to these entities that are involved in running our schemes, there are also an offline Key Distribution Center (KDC). The KDC's role is to distribute the public parameters and credentials of our cryptosystems used in the paper. It also issues public key certificates to the charging stations and EVs that participate in the energy trading and the FI that runs the anonymous payment system. In practice, the KDC can be run by a governmental agency which is interested in the security of the energy trading system [22]. The FI sells *untraceable* digital coins to EVs and exchanges these coins for real currency.

As shown in Fig. 1, in the CS2V scheme, there are charging station(s) owned by private companies that charge EVs. The charging stations can have many charging points that can charge EVs simultaneously, or they can have a few charging points installed in parking lots of clinic, shopping malls, work places, etc. Each EV interacts with the system through a Web or mobile application. The blockchain network in the CS2V is a private network made by the charging stations that run our scheme by processing all transactions' messages sent by the different entities. In private Blockchains, the blockchain enforces strict membership. More specifically, there is an access control mechanism to determine who can join the system, so, every node is authenticated and its identity is known to the other nodes. For efficiency, we adopt Proof-of-Stack (PoS) as a consensus algorithm in the CS2V. The PoS algorithm is based on the idea that the creator of the next block should be chosen via various combinations of random selection, his stake, and age which can provide good scalability. Selected node for making the next block is chosen through a quasi-random process in which the selection depends on assets stored in the wallet (or pool of shares) relating to that node. PoS does not need high computing power to achieve consensus thus it is energy efficient.

As shown in Fig. 2, in the V2V scheme, EVs can be charging vehicles which seek to buy energy or discharging vehicles which have excessive energy and seek to sell it. Unlike the CS2V scheme that uses private blockchain, public blockchains such as Ethereum [23] are used in the V2V scheme. This is because in the V2V scheme, there is no available charging stations or other dedicated entities that can form a private blockchain. In public blockchains, the idea is that each node broadcasts a set of transactions it wants to perform. Special nodes, called miners, collect transactions into blocks, check for their validity, and start a consensus protocol to append the blocks onto the blockchain. Proof-of-work is widely adopted for consensus, where only the miner which can successfully solve a computationally hard puzzle (finding the right nonce for the block header) can append the block to the blockchain.

B. Adversary and Threat Model

Security threats in the energy trading system can come from both internal and external adversaries [24]. In this paper, we assume that the entities (FI, CSs, EVs, and validators/miners) are honest-but-curious, i.e., they execute our schemes correctly, but aim to infer sensitive information about the EVs' drivers by examining the exchanged messages/transactions. We also consider external adversaries that have access to the blockchain and may try to impersonate legitimate users. These adversaries can eavesdrop on the exchanged messages and recorded transactions on the blockchain so that they try to learn the visited locations and the driving patterns of victim EVs' drivers, guess the locations of drivers at a specific time or even track them over time. In addition, attackers may launch Sybil attacks to pretend as multiple non-existing EVs to launch severe attacks such as DoS attacks by submitting many reservations to deprive honest EVs from getting charged or selling their energy which makes the energy trading service unavailable. We also consider attacks against the payment system. Specifically, some attackers may try to create fake coins and double-spend valid coins. Finally, we assume that all parties' devices run in polynomial time and are, thus, unable to break the cryptosystems used in our schemes.

C. Design Goals

Our schemes should achieve the following important objectives.

- Resilience/security. The proposed schemes should not rely on a central entity to run the energy trading system. Central entities are vulnerable of single point of failure and attack.
- *Privacy preservation.* The EVs drivers' privacy is protected by achieving the following two requirements. (i) *Anonymity:* All entities including the CSs, blockchain, and FI should not be able to identify the real identity of an EV that sends a message or makes a transaction in the energy trading system. (ii) *Unlinkability:* Nobody can link two messages sent from the same EV at different times.
- *Anonymity-yet-resistance to Sybil attack.* Launching Sybil attacks by pretending as multiple non-exiting EVs to launch severe attacks such as DoS by submitting a large number of offers/requests pretending that they are submitted from different entities should be thwarted.
- *Authentication.* EVs should be authenticated in an anonymous way so that no adversary can impersonate a legitimate EV.
- *Efficient anonymous payment system.* The payment system should be anonymous and secure against fake payments and double spending attacks. It should also be scalable and efficient.

IV. Proposed Privacy-Preserving CS2V and V2V Energy Trading Schemes

In this section, we describe in detail our privacy-preserving energy trading schemes. We first describe the system initialization phase, followed by the purchase of digital coins phase. Then, we describe in details the CS2V scheme. Finally, we describe the V2V scheme. Table I gives the main notations used in the paper.

A. System Initialization

In this phase, the KDC issues public key certificates to the CSs, EVs, and the FI involved in the anonymous payment system. The KDC first generates a master public/private key pair (mpk/msk) which is used for a digital signature scheme, such as RSA, for signing the certificates. Then, an EV v_i, having a unique identity (e.g., license plate number), creates a public/private key pair (PK_{v_i}/SK_{v_i}) and KDC generates a certificate $cert_{v_i}$

TABLE I
MAIN NOTATIONS

Notation	Description
v_i	An EV
CS	Charging station
FI	Financial institution
dv_j	Discharging EV
cv_i	Charging EV
mpk	Master public key
msk	Master secret key
PK_{v_i}	EV public key
SK_{v_i}	EV private key
$cert_{v_i}$	EV public key certificate
PP	Public parameters of zk-SNARK
m	Number of digital coins
$\{s_1, s_2, \ldots, s_m\}$	Random elements chosen by EV
CP	Coin public key
ID_v	EV real identity
$\{k_1, k_2, \ldots, k_m\}$	Random elements chosen by FI
d	FI secret key
$Sig_{FI}(CP_i)$	FI signature on CP_i
v	Coin's monetary value
TS	Charging time slot
bID_i	Bid ID
PK_i	CS public key
ℓ_i	CS location
CR_i	Charging rate
PO_i	Charging amount

binding PK_{v_i} to v_i. The KDC also runs the *Setup* algorithm of zk-SNARK to generate the public parameters (PP) of the zk-SNARK. Finally, the KDC publishes the zk-SNARK parameters PP and the master public key mpk to the system users. These parameters are used by the EVs to authenticate messages using the common-prefix linkable anonymous authentication scheme. Note that the system initialization phase is done off-line and only once.

B. Purchasing Digital Coins

In our schemes, EVs need to purchase untraceable coins from the FI. This process is performed *outside* the blockchain. Specifically, we assume that the energy trading application on the EV driver's smart phone contains an *e-wallet* service, which stores coins that are digitally signed by the FI. The FI only issues coins with predefined momentary values so that the coin value cannot be used to identify the EV that bought the coin because many EVs buy coins with the same value. The purchase of digital coins proceeds as follows.

1) Suppose the EV (client) wants to purchase m digital coins from the FI. After the payment is made (e.g., through a credit card), the client chooses m secret and random *elements* $\{s_1, s_2, \ldots, s_m\} \in \mathbb{Z}_n^*$ and computes m coin public keys $\{CP_1, CP_2, \ldots, CP_m\}$, where $CP_i = s_i \cdot G$, $\forall i \in \{1, 2, \ldots, m\}$. Each public key uniquely identifies one coin.

2) The client and the FI invoke the blind signature protocol discussed in Section II, so that the client obtains a valid signature $sig_{FI}(CP_i)$ for each purchased coin CP_i without revealing CP_i to FI as follows.

Step 1: The EV sends a purchase request message msg_1

$$msg_1 = ID_v \| M \| Sig_v(ID_v \| M)$$

where ID_v is the EV real identity and M is the number of digital coins the EV wants to buy and their momentary values and $Sig_v(ID_v \| M)$ is the signature on the whole message using the EV private key.

Step 2: The FI chooses m secret and random *elements* $\{k_1, k_2, \ldots, k_m\} \in \mathbb{Z}_n^*$ and computes $R = \{R_1, R_2, \ldots, R_m\}$, where $R_i = k_i \cdot G$, $\forall i \in \{1, 2, \ldots, m\}$. Then, it sends msg_2 to the EV

$$msg_2 = R \| Sig_{FI}(R)$$

Step 3: The EV computes $A_i = R_i + \gamma_i{\cdot}G + \delta_i{\cdot}P$, $\forall i \in \{1, 2, \ldots, m\}$, where $\gamma_i, \delta_i \in \mathbb{Z}_n^*$ are random elements and computes $t_i = x_i \bmod n$, where x_i is the x-coordinate of point A_i. The EV computes $c'_i = (c_i - \delta_i) \bmod n$, where $c_i = H(CP_i || t_i) \bmod n$. The EV sends msg_3 to the FI

$$msg_3 = \{c'_1, c'_2, \ldots, c'_m\}$$

Step 4: The FI computes $s'_i = (k_i$ - $c'_i{\cdot}d) \bmod n$ where d is the FI secret key and sends to the EV msg_4, where,

$$msg_4 = \{s'_1, s'_2, \ldots, s'_m\}$$

Finally, the EV computes $s_i = (s'_i + \gamma_i) \bmod n$ and stores the signature on CP_i ($Sig_{FI}(CP_i)$) as (s_i, c_i). The FI remains oblivious to the values (i.e., public keys) of the individual coins. Note that, in this phase, the client uses his real identity to purchase coins and because of using the blind signatures, the FI cannot link a coin to the EV that bought it when the coin is deposited by a CS and this is required to preserve location privacy.

3) To improve the efficiency of our system, we can allow digital coins of different denominations (e.g., \$1, \$5, \$10). In this case, the FI signs each denomination with a different private key dedicated to this denomination, and each coin is stored as $\langle v, CP_i, sig_{FI^{(v)}}(CP_i)\rangle$, where v is the coin's monetary value, $sig_{FI^{(v)}}(CP_i)$ is the signature of the FI on CP_i using its private key dedicated for the denomination value v. This approach is not prone to privacy leaks by linking a coin value to the client who bought it because coins with the same domination value are bought by many EVs.

C. Privacy-Preserving CS2V Energy Trading Scheme

As illustrated in Fig. 1, the CS2V scheme consists of the following phases. In the *submitting bids* phase, CSs submit competitive charging bids to the blockchain. In the *reservation* phase, an EV that needs to charge submits a reservation request to the blockchain, and the validators verify the reservation request and store it in the blockchain. In the *charging and payment* phase, the EV charges and pays. Finally, the last phase is the *coin deposit* where the CS deposits the coins in the FI. The exchanged messages in CS2V scheme are shown in Fig. 3.

1) Submitting Bids: In this phase, the CSs publish their energy bids on the blockchain as follows. First, the day is divided into predefined timeslots, e.g., each timeslot can be 30 minutes or 1 h. Then, when a CS i has an available charging point, it

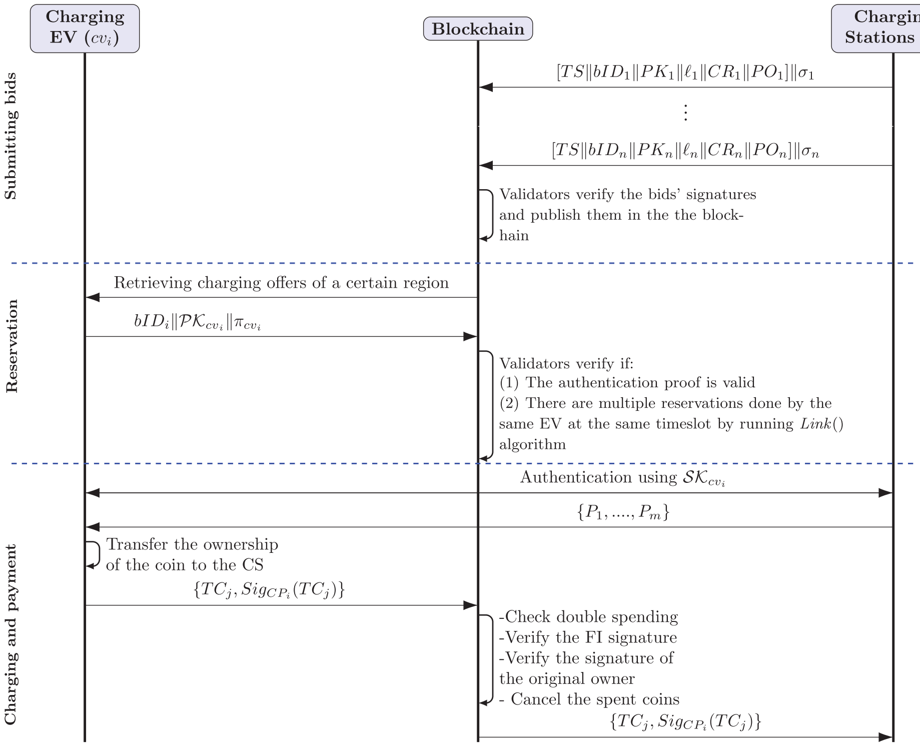

Charging EV (cv_i)
Blockchain
Charging Stations (i)
Submitting bids
$[TS\|bID_1\|PK_1\|\ell_1\|CR_1\|PO_1]\|\sigma_1$
⋮
$[TS\|bID_n\|PK_n\|\ell_n\|CR_n\|PO_n]\|\sigma_n$
Validators verify the bids' signatures and publish them in the the blockchain
Reservation
Retrieving charging offers of a certain region
$bID_i\|\mathcal{PK}_{cv_i}\|\pi_{cv_i}$
Validators verify if:
(1) The authentication proof is valid
(2) There are multiple reservations done by the same EV at the same timeslot by running *Link*() algorithm
Charging and payment
Authentication using $\mathcal{SK}_{cv_i}$
$\{P_1, ..., P_m\}$
Transfer the ownership of the coin to the CS
$\{TC_j, Sig_{CP_i}(TC_j)\}$
-Check double spending
-Verify the FI signature
-Verify the signature of the original owner
- Cancel the spent coins
$\{TC_j, Sig_{CP_i}(TC_j)\}$

Fig. 3. Illustration of the exchanged messages in our CS2V energy trading scheme.

composes a *bidding* message msg_5:

$$msg_5 = TS\|bID_i\|PK_i\|\ell_i\|CR_i\|PO_i,$$

where TS is the timeslot of charging, bID_i is bid ID, PK_i is CS's public key, ℓ_i is the location of the CS, CR_i is the charging rate (price of each KWh), and PO_i is the amount of charging energy (KWh) it can provide.

The message is signed with the private key of the CS and is broadcasted on the blockchain network. Note that a charging station can send multiple bids at a specific timeslot and the number of bids depends on the number of EVs the charging station can charge at a certain timeslot. Before storing the bid on the shared ledger, the validators of the blockchain network should verify the signature of the message.

2) Reservation: In this phase, a charging EV cv_i that needs to charge queries the blockchain to retrieve the bids of the charging stations of a certain geographic area. Then, the EV should select the most appropriate bid in terms of distance to the CS, charging rate, and the amount of energy. The selection of best bid depends on the preference of the charging vehicle. For instance, an EV may prefer a nearby CS regardless of the charging rate while others may prefer lower charging rates.

Once cv_i determines the best bid, it then creates a new public/private key ($\mathcal{PK}_{cv_i}/\mathcal{SK}_{cv_i}$) used for once and the corresponding address $\mathcal{AD}_{cv_i}$. Then, it uses common-prefix-linkable anonymous authentication scheme to generate an attestation π_{cv_i} by running $Auth$ algorithm descried in Section II-D and using the zk-SNARK's public parameters PP and the timeslot TS included in the bid as the prefix as follows:

$$\pi_{cv_i} = Auth(TS||M_{cv_i}, PK_{cv_i}, SK_{cv_i}, cert_{cv_i}, PP),$$

where $M_{cv_i} = bID_j\|\mathcal{PK}_{cv_i}$. Then, cv_i creates a *reservation* message msg_6:

$$msg_6 = bID_j\|\mathcal{PK}_{cv_i}\|\pi_{cv_i}$$

where bID_j is the selected bid ID, π_{cv_i} is the authentication proof and $\mathcal{PK}_{cv_i} = k \cdot G$ is the one-time public key computed by the cv_i and $k \in \mathbb{Z}_n^*$. Then, the message is broadcasted on the blockchain network. Note that $\mathcal{PK}_{cv_i}$ will be used later by the cv_i to prove to the CS that it is the one that indeed made the reservation.

Then, the reservation request is verified and stored in the blockchain. To do so, the validators verify the attestation proof π_{cv_i} by running *Verify* algorithm of common-prefix-linkable anonymous authentication scheme described in Section II-D as follows:

$$Verify(TS||M_{cv_i}, \pi_{cv_i}, mpk, PP),$$

Also, the validators check for multiple reservations by executing $Link(\pi_{cv_i}, \pi_{cv_*})$ discussed in Section II-D for each valid authentication attestation π_{cv_*} that was received in the timeslot TS. The objective here is that it is not allowed that one EV makes more than one reservation in one timeslot. Once all these verifications succeed, the reservation transaction is stored on the ledger. Note that running the *Link()* algorithm is very efficient because it does not require cryptographic operations from the validators and it is just checking the equality of the two

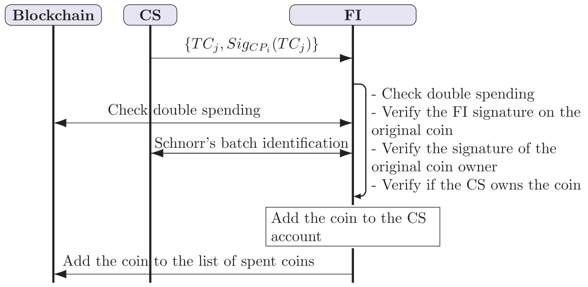

Blockchain
CS
FI
$\{TC_j, Sig_{CP_i}(TC_j)\}$
- Check double spending
- Verify the FI signature on the original coin
- Verify the signature of the original coin owner
- Verify if the CS owns the coin
Check double spending
Schnorr's batch identification
Add the coin to the CS account
Add the coin to the list of spent coins

Fig. 4. Coin deposit procedure.

authentication attestations as described in Section II-D, and this makes our scheme scalable.

3) EV Charging and Payment: After reserving a charging point, in this phase, an EV charges and pays. First, when the EV arrives at the CS, it has to prove that it has reserved a charging point. In particular, the EV and the CS engage in an instance of Schnorr's identification protocol (Section II), in order for the EV to prove to the CS that it knows the private key corresponding to the public key that made the reservation. Following a successful authentication, the EV charges the amount of power in the reservation and then initiates the payment procedure.

The CS chooses m secret random *elements* $\{k_1, k_2, \ldots, k_m\} \in \mathbb{Z}_n^*$ and computes m public keys $\{P_1, P_2, \ldots, P_m\}$, where $P_j = k_j \cdot G$, $\forall j \in \{1, 2, \ldots, m\}$ assuming that cv_i pays m coins to CS. The EV transfers the ownership of each coin to the CS as follows.

The EV first selects a random element $a \in \mathbb{Z}_n^*$, then it computes $a \cdot G$, $r = x_1 \bmod n$, where x_1 is the x-coordinate of the point $a \cdot G$. Then, it computes $a^{-1} \bmod n$ and $b = H(TC_j)$, where $TC_j = CP_i \| sig_{FI}(CP_i) \| P_j$, $CP_i \| sig_{FI}(CP_i)$ is the coin to be transfered, and $H : \{\}^* \rightarrow Z_n^*$. Then, it uses the secret key s_i of the coin $CP_i \| sig_{FI}(CP_i)$ to compute $s = a^{-1}(b + s_i \cdot r) \bmod n$. The value (s, r) is the signature $Sig_{CP_i}(TC_j)$ on TC_j. Thus, $TC_j \| Sig_{CP_i}(TC_j)$ is the transferred coin. Then, the EV sends the transferred coin to the blockchain. The blockchain should do the following: (i) verify the FI signature on the original coin to be transfered, (ii) check if the original coin has not been spent before by checking the list of spent coins on the blockchain and (iii) verify the signature of the transferred coin as follows. It computes $b = H(TC_j)$, $u_1 = b \cdot s^{-1} \bmod n$, and $u_2 = r \cdot s^{-1} \bmod n$. Then, it computes $(x_1, y_1) = u_1 \cdot G + u_2 \cdot CP_j$, if $x_1 = r$, then the signature is valid. If all the verifications succeed, the original coin is stored in the list of spent coins to prevent double spending the coin and the blockchain send transferred coins to the CS.

4) Coin Deposit: In this phase, the CS deposits the coins it collects in a period of time (e.g., several days) in the FI. Fig. 4 illustrates the exchanged messages in this phase. The CS sends a group of coins to the FI. The FI verifies the following for each coin: (i) the coin has not been spent before by checking if the hash of the coin is in the list of spent coins, (ii) the FI signature of the original coin is valid, (iii) the signature that transfers the ownership of the coin from an EV to the CS is valid, and (iv) finally, FI checks if the CS owns the coins; to do so, the CS and the FI should invoke the batch version of Schnorr's protocol described in Section II-B so that the CS can prove that it knows the underlying secrets of the submitted coins. If all these verifications succeed, the FI deposits the coins to the CS account and the CS can exchange coins with real currency if it wants. Finally, the FI sends the coins to the blockchain to be stored in the list of spent coins to avoid re-depositing or re-spending them.

D. Privacy-Preserving V2V Energy Trading Scheme

As illustrated in Fig. 2, the V2V scheme consists of the following phases. In the *charging request* phase, a charging EV submits a charging request to the blockchain containing cloaked location, cloaked time of charging, and an anonymous signature. In the *charging bids* phase, the discharging EVs submit authenticated and anonymous bids to the blockchain. In the *reservation* phase, the charging vehicle selects the best bid. Then, in the *charging and payment* phase, the EV charges and pays. Finally, the last step is the *coin deposit* where the discharging EV deposits the coins in the FI. An illustration to the exchanged messages in the V2V scheme is shown in Fig. 5.

1) Charging Request: In this phase, a charging vehicle cv_i submits an anonymous and authenticated charging request to the blockchain so that the discharging EVs submit their bids. The smart contract given in Algorithm 1 is responsible for executing the V2V energy trading scheme by receiving charging requests from charging EVs and bids from discharging EVs.

To preserve location privacy, let the area $\mathcal{A}$ (e.g., a city), where energy is traded, is divided into a set of regions $\mathcal{R}$ (small geographic areas). Then, cv_i composes a charging request by selecting a cloaked location, i.e., geographic region $\mathcal{R}_{cv_i}$, instead of exact location. It also cloaks the desired time of charging $\mathcal{T}_{cv_i} \in \mathcal{P}(\mathsf{T})$, where $\mathcal{P}(\mathsf{T})$ is a set of time intervals. Next, cv_i chooses a random value $a_i \in \mathbb{Z}_n^*$ and computes g^{a_i}.

Then, cv_i uses common-prefix-linkable anonymous authentication scheme to generate an attestation π_{cv_i} by running $Auth$ algorithm using the zk-SNARK's public parameters PP and the timeslot (TS) as the prefix as follows;

$$\pi_{cv_i} = Auth(TS || M_{cv_i}, PK_{cv_i}, SK_{cv_i}, cert_{cv_i}, PP),$$

where $M_{cv_i} = \mathcal{R}_{cv_i}$, $\mathcal{T}_{cv_i}$, g^{a_i}. The cv_i sends the tuple $\mathcal{R}_{cv_i}$, $\mathcal{T}_{cv_i}$, g^{a_i} and π_{cv_i} to the smart contract given in Algorithm 1. Then, the contract $\mathcal{C}$ verifies the proof π_{cv_i} (see line 6-10 in Algorithm 1) by running:

$$Verify(TS || M_{cv_i}, \pi_{cv_i}, mpk, PP).$$

We should also note that $\mathcal{C}$ ensures that each charging EV cannot submit multiple requests at the same timeslot by executing $Link(\pi_{cv_i}, \pi_{cv_*})$ for each valid authentication attestation π_{cv_*} that was received and stored in the smart contract at a specific timeslot (see line 5-11 in Algorithm 1).

2) Charging Bids: In this phase, discharging vehicles interested in a certain energy request send their authenticated and encrypted bids to the smart contract as follows. dv_j checks available charging requests on the blockchain and checks whether its location ℓ_{dv_j} lies in the region of the request $\mathcal{R}_{cv_i}$. Then, it chooses a random value $b_j \in \mathbb{Z}_n^*$ and computes g^{b_j}. Then, dv_j

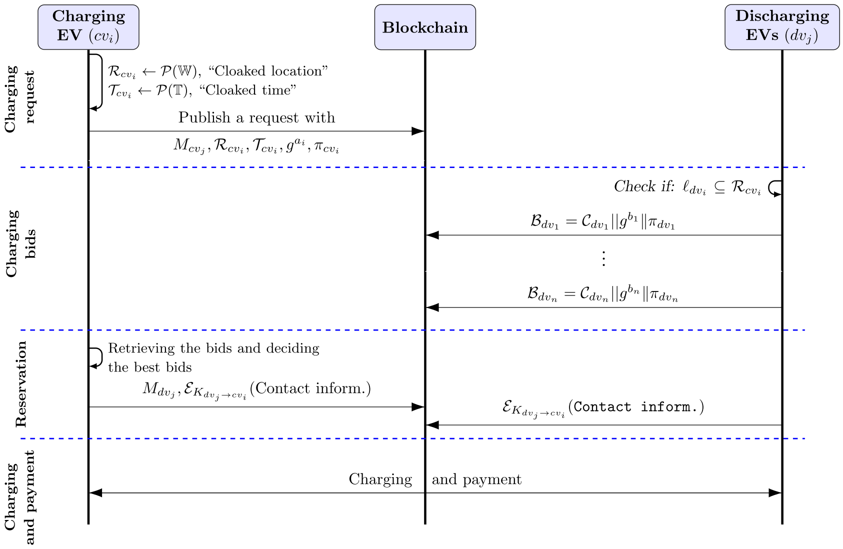

Charging EV (cv_i)
Blockchain
Discharging EVs (dv_j)
Charging request
$\mathcal{R}_{cv_i} \leftarrow \mathcal{P}(\mathbb{W})$, "Cloaked location"
$\mathcal{T}_{cv_i} \leftarrow \mathcal{P}(\mathbb{T})$, "Cloaked time"
Publish a request with
$M_{cv_j}, \mathcal{R}_{cv_i}, \mathcal{T}_{cv_i}, g^{a_i}, \pi_{cv_i}$
Charging bids
Check if: $\ell_{dv_i} \subseteq \mathcal{R}_{cv_i}$
$\mathcal{B}_{dv_1} = \mathcal{C}_{dv_1} || g^{b_1} \| \pi_{dv_1}$
$\vdots$
$\mathcal{B}_{dv_n} = \mathcal{C}_{dv_n} || g^{b_n} \| \pi_{dv_n}$
Reservation
Retrieving the bids and deciding the best bids
$M_{dv_j}, \mathcal{E}_{K_{dv_j \rightarrow cv_i}}$(Contact inform.)
$\mathcal{E}_{K_{dv_j \rightarrow cv_i}}$(Contact inform.)
Charging and payment
Charging and payment

Fig. 5. The V2V energy trading scheme. Note that $\mathcal{E}$ denotes a symmetric key encryption algorithm, e.g., AES-128.

Algorithm 1: Pseudocode of the V2V Energy Erading Contract.

```
contract EnergyTrading
    PP // SNARKs public parameters and KDC's master
        public key
    mapping(address => string) Proofs // Mapping for
        received attestations
    mapping(address => string) Reserved DVs // Mapping
        for reserved discharging EVs
    function EnergyTrading($\mathcal{R}_{cv_i}$, $\mathcal{T}_{cv_i}$, g$^{a_i}$,$\pi_{cv_i}$ )
    if $Verify(TS \| M_{cv_i}, \pi_{cv_i}, mpk, PP)$!=1
        return // Unauthenticated request
    else
        $\forall \pi_* \in$ Proofs $checkif$ $Link(\pi_{cv_i}, \pi_*)! = 1$
        Proofs[]$\leftarrow$ $\pi_{cv_i}$; // store $\pi_{cv_i}$ in the proofs[]
            if this is the first charging request
    end
    function RecieveOffer($\mathcal{B}_{dv_j}$)
        address [] DVlist // Received discharging EVs'
            offers
        address [] DVProofs // Received discharging
            EVs' proofs
        if $Verify(TS \| M_{dv_j}, \pi_{dv_j}, mpk, PP)$!=1
            return // Unauthenticated offer
        else
            $\forall \pi_* \in$ DVlist[] $checkif$ $Link(\pi_{dv_j}, \pi_*)! = 1$
            DVProofs[]$\leftarrow$ $\pi_{dv_j}$;
            DVlist[]$\leftarrow$dv$_j$;
            // check if the $dv_j$ has sent multiple bids
                for the same EV
        end
    function Reservation($M_{dv_j}$)
        receive the selected addresses of discharging
         vehicles.
        Reserved DVs[]$\leftarrow$ $\pi_{dv_j}$;
```

uses g^{a_i} included in the charging request to compute a symmetric key $k_{dv_j \rightarrow cv_i}$ as follows:

$$k_{dv_j \rightarrow cv_i} = (g^{a_i})^{b_j} = g^{a_i b_j}$$

Then, it encrypts the tuple (PO_{dv_j}, CR_{dv_j}) with $k_{dv_j \rightarrow cv_i}$ using a symmetric key encryption algorithm, e.g., AES-128, to obtain the ciphertext $C_{dv_j} = \mathcal{E}_{k_{dv_j \rightarrow cv_i}}[PO_{dv_j}, CR_{dv_j}]$, where PO_{dv_j} is the amount of energy that dv_j can sell and CR_{dv_j} is the charging rate (price/kWh). Note that we use symmetric key encryption since it is much efficient than asymmetric key encryption in terms of computation and communication overheads. Then, dv_j uses common-prefix-linkable anonymous authentication scheme to generate an attestation π_{dv_j} by running $Auth$ algorithm and TS as the prefix as follows:

$$\pi_{dv_j} = Auth(TS||M_{dv_j}, PK_{dv_j}, SK_{dv_j}, cert_{dv_j}, PP),$$

where $M_{dv_j} = C_{dv_j}||g^{b_j}$. Then, it sends its bid $\mathcal{B}_{dv_j} = \mathcal{C}_{dv_j}||g^{b_j}\|\pi_{dv_j}$ to the smart contract. Note that π_{dv_j} is used so that the contract can check if the discharging vehicle dv_j has not already been reserved in one timeslot more than once. Also, discharging EVs are allowed to send multiple bids in the same timeslots for different requests because by sending only one bid it is not guaranteed that it will be selected, however the smart contract prevents reserving the same discharging EV for more than one charging EV.

3) Reservation: In the reservation phase, once the smart contract collects the bids from the discharging vehicles, the charging EV decides the best bid and makes the reservation. To do that, cv_i first retrieves the bids from the blockchain, and then, it computes a symmetric key shared with each dv_j that has sent a bid. For example, cv_i uses g^{b_j} included in the bid of dv_j to compute a symmetric key $k_{cv_i \rightarrow dv_j}$ as follows:

$$k_{cv_i \rightarrow dv_j} = (g^{b_j})^{a_i} = g^{b_j a_i}$$

Then, cv_i decrypts all bids and selects the best bid based on the price and the amount of energy. Finally, cv_i sends a transaction to the smart contract that contains the IDs of the selected bids and the contact information, i.e., email address or phone number of the charging EV, encrypted with the shared secret key. Note that g^{b_j} can be used as the ID of the bids. Finally, the selected discharging vehicle sends its contact information encrypted by the shared symmetric key with the charging vehicle to the contract.

4) EV Charging and Payment: In this phase, the charging EV meets the discharging EV to charge and pay. When they meet, they should first authenticate each other using the shared

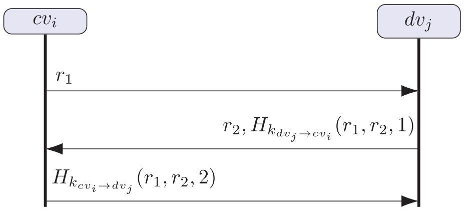

cv_i
dv_j
r_1
$r_2, H_{k_{dv_j \to cv_i}}(r_1, r_2, 1)$
$H_{k_{cv_i \to dv_j}}(r_1, r_2, 2)$

Fig. 6. Mutual authentication between charging and discharging EVs.

secret key. To do that, as shown in Fig. 6, cv_i first sends a random challenge r_1, then dv_j chooses a random challenge r_2 and computes the keyed hash of the tuple $(r_1, r_2, 1)$ using the shared secret key $k_{dv_j \to cv_i}$ and sends it back to cv_i. Then, using the shared secret key $k_{cv_i \to dv_j}$, cv_i verifies the keyed hash function sent from dv_j and computes the keyed hash value $H_{k_{cv_i \to dv_j}}(r_1, r_2, 2)$ and sends it back to the dv_j that verifies this hash value. After charging, cv_i pays dv_i using the payment procedure explained in Section IV-C3.

5) Coin Deposit: Finally, the last step is the *coin deposit* where the discharging EV deposits the coins it collects in a period of time (e.g., several days) in the FI. The payment deposit procedure explained in subsection IV-C4 can be used by the discharging EV to deposit the coins.

V. Evaluations

In this section, we first discuss the security and privacy analysis of our schemes followed by the performance evaluations.

A. Security and Privacy Analysis

In this section, we discuss how our schemes achieve the security/privacy objectives discussed in subsection III-C.

Security and transparency. Our schemes are secure against single point of failure and attack due to using blockchain (a distributed network) to run them instead of using a central server. Moreover, due to the immutability nature of the blockchain, the data stored on the blockchain cannot be tampered with by malicious adversaries. Also, our schemes ensure *transparency* of the trading system. This is because unlike centralized systems in which it is not known how the central unit runs the schemes, so it can favor charging stations over others to sell their offers first, in our schemes, all transactions are posted on the blockchain and the CSs and EVs can ensure that their bids/requests are received and handled properly.

Preserving the privacy of EVs' drivers. In our schemes, we have used several techniques to preserve the privacy of EVs' drivers such as cloaking technique, anonymous authentication scheme, one-time public/private key pair, and anonymous payment. To ensure anonymity of EVs' drivers, the EV drivers use a random blockchain address that acts as a *pseudonym* and the generation of that address is unlinakable to the EV driver's real identity. Moreover, unlike the CS2V scheme, the private information of both the charging and discharging vehicles (location, time, and amount of energy) should be protected in the V2V scheme. This is achieved by (i) cloaking the location of the charging EV as well as the time of trading, and (ii) encrypting the submitted bids using a symmetric key encryption shared between charging and discharging EVs; therefore, the bids are *confidential* to everyone including the blockchain miners and other attackers. Also, the underlying common-prefix-linkable anonymous authentication scheme ensures that an EV can authenticate messages without being able to identify the sender of the messages or link them. Finally, the anonymous payment system ensures that the coins used for payment are untraceable and are not linked to a specific EV.

Unlinkabilty. In our schemes, no one can link a charging reservation or bid to a specific EV that sent it. This is because EVs interact with the blockchain with a randomly generated blockchain address. The blockchain address is a one-time *pseudonym* generated by the EVs, and it cannot reveal the EV's real identity. Moreover, the underlying common-prefix-linkable anonymous authentication scheme ensures that an EV can authenticate messages without being linked if an EV sends one message with each prefix so the EV is anonymous and no one can link its messages. Also, in the V2V scheme, a random value a_i should change each time an EV sends a charging request so that no one can know if the messages are sent from the same EV or not.

Anonymous and secure payment system. In our schemes, blockchain is used for storing the hash values of spent coins so that double spending can be detected. Thus, due to the use of blockchain, all spent coins are stored in tamper-resistant and non-repudiable ledger. In addition, once the coin's ownership is transferred, no one can re-spend or re-transfer the same coin because the spent coins are stored in the blockchain. Moreover, during the purchase of digital coins, the EV's real identity is used but because of using the blind signature scheme, the FI cannot link the coins it signed to the EV that bought them. The payment is also secure against stolen coins. This is because if an attacker steals coins, he/she cannot use them in another transactions because the spender of a digital coin has to prove with a ZKP protocol that it knows the coin's private key without revealing it.

Anonymity-yet-resistance to Sybil attacks. Our schemes ensure the anonymity of EVs drivers during the energy trading because the real identity of each vehicle is not used and no one can link the messages sent from the same EV. However, a malicious EV may try to launch a Sybil attack by pretending as multiple non-existing EVs to launch more powerful attacks such as DoS on the trading system by submitting a large number of reservations/requests to the blockchain to make the system unreliable and the service is unavailable. In our schemes, the blockchain can detect such submissions using the linkable anonymous authentication scheme. This is because, for an EV to successfully authenticate messages, it needs to include a *prefix* to them. If an EV authenticates reservations/requests more than once for the same transaction (i.e, with the same prefix), the blockchain can link such submissions. In this way, Sybil submissions can be discarded to ensure that the system is reliable and the service is available.

Mutual authentication. In our schemes, an EV, either charging or discharging, needs to authenticate messages using their secret credentials by running the *Auth* algorithm of the common linkable anonymous authentication scheme. In addition, before starting the energy trading, a charging EV needs to prove to the CS (or the discharging EV) that it is the one that indeed made the reservation. In the CS2V, the EV and the CS engage in an instance of Schnorr's identification protocol in order for the EV to prove to the CS that it knows the private key corresponding to the public key that is computed by the EV that made the reservation. In the V2V scheme, only charging/discharging EVs which share the same symmetric secret key can successfully authenticate each other. Specifically, by verifying the keyed hash value $H_{k_{dv_j \rightarrow cv_i}}(r_1, r_2, 1)$, the cv_i can make sure that dv_j knows the shared secret key. Similarly, by verifying $H_{k_{cv_i \rightarrow dv_j}}(r_1, r_2, 2)$, dv_j can make sure that cv_i knows the shared key. This is required to ensure that no adversary can impersonate legitimate users. Moreover, the man-in-the-middle attack is impossible without the need for additional signatures. This is because any attempt from the miners to change the value g^{a_i} (or g^{b_j}) to launch man-in-the-middle attack can be detected because the blockchain is transparent and the ledger is immutable, so EVs can ensure that g^{a_i} (or g^{b_j}) they sent is received and stored correctly.

Our schemes thwart collusion attacks that aim to obtain private information about the EVs. The FI may collude with other EVs by exchanging the messages with an EV during the purchasing digital coins phase to get private information about the EVs such as when they charge from a particular EV. However, by using PBS during purchasing digital coins phase, only the requester of the coins can unblind the FI signature to obtain the anonymous coins. Thus, an attacker EV who has a set of exchanged messages with the FI can not link them with a particular EV.

B. Performance Evaluation

In this section, we evaluate the performance of the proposed schemes in terms of communication and computation overheads.

1) Experiment Setup: We have implemented all cryptosystems used in our schemes with the C programming language, using OpenSSL's library for elliptic curve cryptography, and a 2.8 GHz Intel Core i7 CPU. For 128-bit security, we have selected ANSI's *X9.62 Prime 256v1* curve, whose order n is a 256-bit prime. We have implemented the proposed CS2V scheme and deployed it in Ganache blockchain [25] platform, which is a private blockchain. We have also implemented the V2V scheme and deployed it in Kovan blockchain which is a testnet of the Ethereum public blockchain [23].

We have used Jsnark library [26] for building the circuits of zk-SNARKs used in the underlying common-prefix linkable anonymous authentication scheme. The library uses libsnark [27] as a backend and has built-in gadgets, i.e., small boolean circuits including a gadget of SHA-256 and gadgets for encryption algorithms such as AES. Then, we wrote the statements of zk-SNARKs in Jsnark using the underlying gadgets. The public parameters of zk-SNARK are generated using the libsnark generator library. Finally, to verify the proofs of the underlying common-prefix linkable anonymous authentication scheme, we have used the modified Ethereum client [28] that is written in Java 1.8 with Spring framework [29].

TABLE II
SIZE OF ELEMENTS OF OUR SCHEMES' MESSAGES

Data	Size (bytes)	Data	Size (bytes)
Timeslot	6	Field element	32
Location	6	Group element	64
Charging rate	2	Public key	64
Amount of energy	2	Authentication proof	729

The metrics that are used in our evaluations are the communication and computation overheads. The computation overhead is defined as the processing time required by each entity in our schemes. The communication overhead is measured by the size of messages transmitted between the entities in bytes. We also measure the gas consumption needed by our V2V scheme since the underlying blockchain is the public blockchain. *Gas* is used to quantify the execution cost associated with each transaction.

2) Communication and Storage Overhead: To measure the communication overhead, our schemes use signature scheme that uses elliptic curve cryptography (ECC). Using an elliptic curve additive group of order 256 bits, the signature's size is 64 bytes [30]. Using Table II that summarizes the size of the elements used in our schemes' messages, we calculate the size of the messages sent in our schemes. Table III summarizes the total communication overhead in our schemes.

Communication overhead of the anonymous payment. To purchase a digital coin, the EV and the FI need to run the blind signature scheme as discussed in Section IV-B. The EV sends msg_1 that contains the number and value of coins it wants to purchase. Also, it sends msg_3 that has one field element. Thus, the size of msg_1 and msg_3 are 70 and 32 bytes respectively. The FI sends msg_2 that has one group point and a signature. It also need to send msg_4 that has one field element. Thus, the size of msg_2 and msg_4 are 128 and 32 bytes, respectively

Then, during the payment phase, the EV sends the transferred coin to the blockchain that contains two public keys, FI's signature, and the EV's signature. Thus, the total size of the transferred coin is 256 bytes.

To deposit the coins that the CS collects in a period of time, it needs to send 256 bytes for each transferred coin to the FI. Thus, to send n coins, the total overhead is $n \times 256$ bytes. Finally, the CS needs to prove to the FI that it owns the coin by proving that it knows the secret that corresponds to the public key in the coin using Schorr's batch protocol. To do so, the CS sends a commitment, i.e., a group element, then the FI sends a challenge, i.e., a field element and the CS sends a response, i.e., a field element. Thus, the total communication overhead is 128 bytes. Fig. 7 evaluates the effect of Schorr's batch protocol on verifying a group of coins instead of individual ones. As described in Section II, the batch protocol involves the transmission of the same size of messages as the original protocol, so the communication cost is constant (128 bytes) with respect to the number of coins. On the other hand, the cost of executing individual instances of Schnorr's protocol incurs a cost that grows linearly with the number of coins.

Communication overhead of the CS2V scheme. In the *submitting bids* phase, a charging station needs to send a bid that

TABLE III
COMMUNICATION OVERHEAD IN OUR SCHEMES

	Entity	Message	Communication overhead "bytes"
Payment	EV	Purchasing digital coin (msg_1 and msg_3)	102
	FI	Purchasing digital coin (msg_2 and msg_4)	160
	EV	Coin transfer	256
	CS	Coin deposit	256
CS2V	CS	Submitting bid	150
	Charging EV	Reservation	795
V2V	Charging EV	Charging request	741
		Reservation	64
	Discharging EV	Charging bid	745

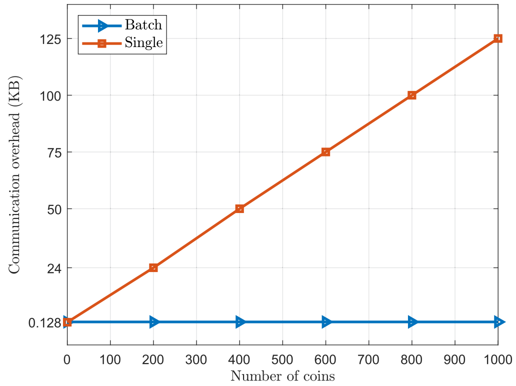


Fig. 7. Effect of Schnorr's batch protocol on coins ownership verification overhead.

TABLE IV
COMPUTATION OVERHEAD OF INDIVIDUAL OPERATIONS

Operation	Time
Multiplication	0.005 ms
Exponentiation	4.4 ms
Signature generation	3 ms
Signature verification	1.5 ms
Encryption (AES-128)	0.0203 ms
Decryption (AES-128)	0.0078 ms
Authentication proof generation	10 s
Authentication proof verification	2.5 ms

contains the following: timeslot, bid ID, location, charging rate, amount of energy, CS's public key and a signature. The total size of the packet is 150 bytes. In the *reservation* phase, a charging EV needs to send a reservation request that contains bid ID, CS's public key and authentication proof. The total size of the packet is 795 bytes. The scheme in [4] is the most relevant work to our CS2V scheme. In this scheme, the CS require 38 bytes in the bidding phase, and the EV needs to send 24 bytes in the reservation phase. Although, the overhead in [4] is lower than our scheme, it does not consider resistance to Sybil attacks, authentication, and payment, as will be explained in the related works section.

Communication overhead of the V2V scheme. In the *charging request* phase, a charging EV needs to send a charging request that has the cloaked region and time and the authentication proof. The total size of the message is 741 bytes. In the *charging bids* phase, the discharging EV sends an offer that has ciphertext of the bid's information and the authentication proof. The total size of the message is 745 bytes. Finally, in the *reservation* phase, the reservation message includes 64 bytes for each of public key of the selected EV.

Storage overhead. For the storage overhead on the blockchain in the CS2V scheme, the blockchain stores the charging bids (150 bytes per each bid) and hashes of the spent coins (256 bytes). Thus, the total storage overhead for storing $\mathcal{M}$ bids and $\mathcal{N}$ spent coins is (150 × $\mathcal{M}$) + (256 × $\mathcal{N}$) bytes.

In the V2V scheme, the blockchain stores the charging requests (741 per each request), bids (745 per each bid) and hash of spent coins (256 bytes). Thus, the total storage overhead for each charging transaction with $\mathcal{M}$ bids and $\mathcal{N}$ spent coins is 741 + (745 × $\mathcal{M}$) + (256 × $\mathcal{N}$) bytes.

3) Computation Overhead: Our measurements indicate that the multiplication and exponentiation operations take 0.005 ms and 4.4 ms, respectively. Note that the addition operation takes a very short time so it can be neglected. For symmetric key encryption using AES-128, the encryption operation takes 0.0203 ms while the decryption operation takes 0.0078 ms. For signature algorithm, the signing operation takes 3 ms while the verification takes 1.5 ms. Using Table IV that summarizes the computation overhead of individual operations, we calculate the computation time required to compose messages in our schemes and summarize the results in Table V.

Anonymous payment computation overhead. To purchase a digital coin, an EV needs to compute two multiplication operations and one signature. Thus, the computation overhead is 3.01 ms. The FI computes two multiplication operations and one signature. Thus, the computation overhead is 3.01 ms.

To transfer a coin, the EV needs to compute a signature that takes 3 ms. Then, the validators verify two signatures, i.e., the FI's signature and the signature of the old coin's owner on transferred coin. Thus, the computation overhead is 6 ms. Then, to deposit a coin, the FI needs 6 ms to verify two signatures on the transferred coin. Finally, the CS needs to prove the ownership of the digital coins using Schnorr's batch protocol. In Fig. 8, we illustrate the efficiency of Schnorr's batch protocol in verifying the ownership of a large number of coins. It can be seen that verifying 1000 coins individually (without Schnorr's batch protocol) requires 125 ms at the prover and 250 ms at the verifier, whereas the batch protocol reduces these times to 0.1 ms and

TABLE V
COMPUTATION OVERHEAD IN OUR SCHEMES

	Entity	**Message**	**Computation overhead**
Payment	EV/FI	Purchasing digital coin	3.01 ms
	EV	Coin transfer	3 ms
	CS/FI	Coin deposit	6 ms
CS2V	Charging station	Submitting bid	3 ms
	Charging EV	Reservation	10 sec
	Blockchain	Reservation	11.31 ms
V2V	Charging EV	Charging request	10 sec
		Reservation	9 ms
	Discharging EV	Charging bid	10 sec

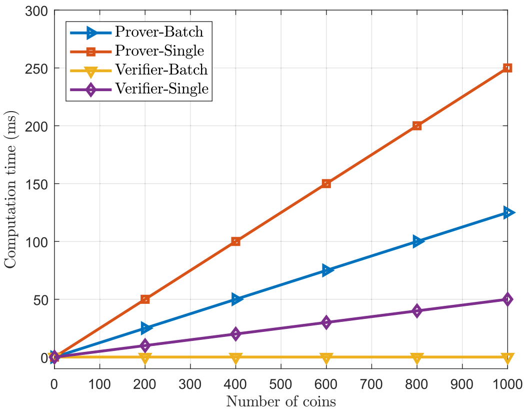


Fig. 8. Effect of Schnorr's batch protocol on computation time of verifying the coins ownership.

TABLE VI
EXECUTION TIME OF VERIFYING THE AUTHENTICATION PROOFS

Number of proofs	**2**	**4**	**6**	**8**	**10**
Time (ms)	10.9	15.5	16.3	17.0	17.9

54 ms, respectively. This indicates that Schnorr's batch protocol makes the payment process fast with less computation overhead.

Computation overhead of the CS2V scheme. In the *submitting bids* phase, the computation overhead on the charging station is 3 ms to compute a signature on the message and 1.5 ms for the validators to verify the signature. In the *reservation* phase, the charging EV computes authentication proof using the common-prefix linkable anonymous authentication scheme that needs 10 seconds. Then, the validators verify the authentication proof that needs 2.5 ms. To evaluate the computation overhead as the number of received authentication proofs increases, Table VI gives the execution time of verifying a number of the authentication proofs. It can be seen that verifying 10 proofs requires 17.9 ms and thus the on-chain computation overhead is in range of *milliseconds which is acceptable*. Although the time needed for the proof generation operation is longer than other operations, sending charging request and bid messages does not need real-time communication because EVs have enough time to compose the messages, and also the verification of the proof which is done by the blockchain needs much shorter time which is important in our application because the proof generation is done by each EV while the validators have to verify a large number of proofs sent by EVs.

TABLE VII
EXECUTION COSTS OF OUR V2V IN ETHERUM

Transaction	**GAS**	**Price (ETH)**
Charging request	76981	0.0003848
Charging bid	89686	0.0004834
Reservation	53981	0.0002699

Computation overhead of the V2V scheme. In the *charging request* phase, the charging vehicle computes one exponentiation operation to obtain g^{a_i} and the authentication proof. Thus, the total time is nearly 10 seconds. In the *charging bids* phase, the discharging EV needs to compute two exponentiation operations, an authentication proof, and an encryption operation to encrypt the bid. Therefore, the total time is 10.04 seconds. In the *reservation* phase, the charging EV computes one exponentiation and one decryption operation to decrypt the bid. The total time is 4.4 ms

Moreover, since the V2V scheme is designed on top of public blockchain, we have also implemented the smart contract of Algorithm 1 in Solidity [31]. Then, it is deployed on the Kovan blockchain which is a testnet of the Ethereum blockchain [23]. In Ethereum, *gas* is used to quantify the execution cost associated with each transaction. The cost is payable using the Ethereum currency, named *Ether*. Table VII summarizes the execution cost associated with each transaction in the V2V energy trading scheme. A charging vehicle needs to submit a charging request that costs 76 K gas, then a discharging vehicle submits a bid that costs 89 K gas, and finally, 65 K gas is required by the charging vehicle to make the reservation. To calculate the transaction fees, according to [32], the cost for 1 unit of GAS is on average 5 Gwei and 1 Ether = 10^9 Gwei. The transaction fees associated with each transaction is given in Table VII. Clearly, the required fees are low for both the charging and discharging EVs.

VI. RELATED WORK

There has been some research work in the areas of EV location privacy, anonymous payment systems, and energy trading. In

TABLE VIII
COMPARISON BETWEEN OUR WORK AND OTHER EXISTING SCHEMES

	Architecture	CS2V	V2V	Privacy	Resistance to Sybil attacks	Authentication	Payment
[4]	Blockchain	√	×	√	×	×	×
[13]	Blockchain	√	×	×	×	×	×
[14]	Blockchain	×	√	×	×	×	×
[15]	Blockchain	×	√	×	×	×	×
[33]	Blockchain	√	×	√	×	×	×
[34]	Blockchain	√	×	×	×	×	×
[35]	Matching Market and Signcryption	√	×	√	×	√	×
Ours	Blockchain	√	√	√	√	√	√

[1]Note: √ denote a realized feature and × denotes an unrealized feature.

Table VIII, we summarize the main differences between existing schemes and ours in terms of desired features such as privacy, resistance to Sybil attacks, authentication, payment, and considered mode of charging.

In [13], the authors use smart contracts to dynamically select the best bids from various charging stations. The EVs send their planned routes and battery status to the blockchain, and then charging stations offer their prices so that the EVs select best offered price. Nevertheless, the proposed scheme does not consider charging reservation, or the underlying payment mechanism. Recently, inspired by bitcoin, a PriWatt system is introduced by [36] that enables a blockchain-based private decentralized energy trading system. The system allows peer-to-peer energy trading without the need for a third-party intermediary.

In [4], charging stations make bids to EVs' drivers in response to their charging requests. To select their preferred charging bids, EVs send hidden commitments to the blockchain. These commitments are used later so that the EV can prove to the charging station that it made the reservation. However, the charging station does not know if the charging point has been reserved or not until the EV drives to it and this may lead to poor management of the service by the CS. In [35], a framework for electric vehicle charging in the IoT infrastructure has been proposed. The framework uses the matching market concept to identify a charging station and uses lattice-based cryptography to maintain security and privacy preservation. However, these papers do not propose an anonymous payment method to protect the privacy of the EVs. Also, they do not consider the Sybil attacks in which attackers may pretend as multiple non-existing EVs and launch severe attacks such as DoS attack by sending a large number of reservation requests to block charging stations from getting customers to use the service.

In [33], the authors proposed a blockchain-based energy trading scheme to supervise and manage the energy trading. It provided anonymous authentication to provide user privacy and fine-grained access control for energy trading services. Moreover, it preserved identity privacy and transaction privacy during energy trading. In [34], the authors proposed a lightweight blockchain-based data sharing and energy trading scheme. A tangle data structure is used to record the transactions in the network in a secure and scalable manner. Also, a game theory model is used to perform negotiation between the grid and vehicles at an optimized cost.

Other schemes have used blockchain for energy trading but they focus on optimization techniques for the electricity pricing and the amount of traded electricity. Kang *et al.* [5] have proposed a consortium blockchain to establish a decentralized electricity trading system for V2V charging. To optimize electricity pricing and the amount of traded electricity among EVs, an optimization technique, called iterative double auction, has been used. In [14], the authors have introduced a permissioned energy blockchain system to implement secure charging services for EVs. The paper uses a Byzantine fault tolerance consensus algorithm to reach the consensus in the energy blockchain. The contract theory has also been used to satisfy the energy needed from EVs while maximizing the operator's profit. Huang *et al.* [15] have proposed an optimal charging scheduling algorithm that considers different vehicle charging scenarios such as V2V and CS2V. A double-objective optimization model has been used to maximize the user's satisfaction and minimizing cost. However these papers do not consider the V2V security, privacy, resistance to Sybil attacks, or authentication. Also, they do not propose an anonymous payment method to protect the privacy of the EVs.

VII. CONCLUSION

In this paper, we have proposed privacy-preserving schemes for CS2V and V2V energy trading. The schemes are built on top of blockchain technology that provides security and transparency without the need for a trusted third party. Moreover, launching Sybil attack by pretending as multiple non-existing EVs to launch powerful attacks such DoS by submitting a large number of reservations/offers is thwarted to ensure that the energy trading system is reliable and the service is available. To preserve the privacy of EV drivers, we have developed anonymous and efficient payment system using blockchain to allow EVs to pay their charging fees with untraceable digital coins. Also, the payment system is secure against stolen coins attack and fake payments. Analysis and experiments are conducted to evaluate the proposed schemes and the results indicate that our schemes are secure and can preserve the privacy of EVs' drivers, and the communication and computation overheads are acceptable.

References

[1] J. Wang, G. R. Bharati, S. Paudyal, O. Ceylan, B. P. Bhattarai, and K. S. Myers, "Coordinated electric vehicle charging with reactive power support to distribution grids," *IEEE Trans. Ind. Informat.*, vol. 15, no. 1, pp. 54–63, Jan. 2019.

[2] D. T. Hoang, P. Wang, D. Niyato, and E. Hossain, "Charging and discharging of plug-in electric vehicles (PEVs) in vehicle-to-grid (V2G) systems: A cyber insurance-based model," *IEEE Access*, vol. 5, pp. 732–754, 2017.

[3] R. Alvaro-Hermana, J. Fraile-Ardanuy, P. J. Zufiria, L. Knapen, and D. Janssens, "Peer to peer energy trading with electric vehicles," *IEEE Intell. Transp. Syst. Mag.*, vol. 8, no. 3, pp. 33–44, Sep.–Nov. 2016.

[4] F. Knirsch, A. Unterweger, and D. Engel, "Privacy-preserving blockchain-based electric vehicle charging with dynamic tariff decisions," *Comput. Sci. - RD*, vol. 33, no. 1–2, pp. 71–79, 2018.

[5] J. Kang, R. Yu, X. Huang, S. Maharjan, Y. Zhang, and E. Hossain, "Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchains," *IEEE Trans. Ind. Informat.*, vol. 13, no. 6, pp. 3154–3164, Dec. 2017.

[6] C. Zhang *et al.*, "BSFP: Blockchain-enabled smart parking with fairness, reliability and privacy protection," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 6578–6591, Jun. 2020.

[7] M. Baza, M. Mahmoud, G. Srivastava, W. Alasmary, and M. Younis, "A light blockchain-powered privacy-preserving organization scheme for ride sharing services," in *Proc. IEEE Veh. Technol. Conf.*, Belgium, 2020, pp. 1–6.

[8] M. Baza, N. Lasla, M. Mahmoud, G. Srivastava, and M. Abdallah, "B-Ride: Ride sharing with privacy-preservation, trust and fair payment atop public blockchain," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 1214–1229, Apr.–Jun. 2021.

[9] New York Times: Uber Settles Data Breach Investigation for $148 Million. [Online]. Available: https://www.nytimes.com/2018/09/26/technology/uber-data-breach.html

[10] E. B. Sasson *et al.*, "Zerocash: Decentralized anonymous payments from bitcoin," in *Proc. IEEE Symp. Secur. Privacy*, 2014, pp. 459–474.

[11] *Countries That Have Banned Cryptocurrency*. [Online]. Available: https://www.escapeartist.com/blog/countries-banned-bitcoin/

[12] W. Al Amiri, M. Baza, K. Banawan, M. Mahmoud, W. Alasmary, and K. Akkaya, "Privacy-preserving smart parking system using blockchain and private information retrieval," in *Proc. Int. Conf. Smart Appl., Commun. Netw.*, 2019, pp. 1–6.

[13] M. Pustisek, A. Kos, and U. Sedlar, "Blockchain based autonomous selection of electric vehicle charging station," in *Proc. Int. Conf. Identification, Inf. Knowl. Internet Things*, 2016, pp. 217–222.

[14] Z. Su, Y. Wang, Q. Xu, M. Fei, Y.-C. Tian, and N. Zhang, "A secure charging scheme for electric vehicles with smart communities in energy blockchain," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4601–4613, Jun. 2019.

[15] X. Huang, Y. Zhang, D. Li, and L. Han, "An optimal scheduling algorithm for hybrid EV charging scenario using consortium blockchains," *Future Gener. Comput. Syst.*, vol. 91, pp. 555–562, 2019.

[16] Y. Lu, Q. Tang, and G. Wang, "ZebraLancer: Private and anonymous crowdsourcing system a top open blockchain," in *Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst.*, 2018, pp. 853–865.

[17] E. M. Radi, N. Lasla, S. Bakiras, and M. Mahmoud, "Privacy-preserving electric vehicle charging for peer-to-peer energy trading ecosystems," in *Proc. IEEE Int. Conf. Commun.*, 2019, pp. 1–6.

[18] Q. ShenTu and J. Yu, "A blind-mixing scheme for Bitcoin based on an elliptic curve cryptography blind digital signature algorithm," *CoRR*, 2015. [Online]. Available: http://arxiv.org/abs/1510.05833

[19] C. Schnorr, "Efficient signature generation by smart cards," *J. Cryptol.*, vol. 4, no. 3, pp. 161–174, 1991.

[20] R. Gennaro, D. Leigh, R. Sundaram, and W. S. Yerazunis, "Batching schnorr identification scheme with applications to privacy-preserving authorization and low-bandwidth communication devices," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2004, pp. 276–292.

[21] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza, "Succinct non-interactive zero knowledge for a von Neumann architecture," in *Proc. USENIX Secur. Symp.*, 2014, pp. 781–796.

[22] Y. Zhou, M. Wang, H. Hao, L. Johnson, and H. Wang, "Plug-in electric vehicle market penetration and incentives: A global review," *Mitigation Adapt. Strategies Glob. Change*, vol. 20, no. 5, pp. 777–795, 2015.

[23] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, 2014.

[24] N. Saxena, S. Grijalva, V. Chukwuka, and A. V. Vasilakos, "Network security and privacy challenges in smart vehicle-to-grid," *IEEE Wireless Commun.*, vol. 24, no. 4, pp. 88–98, Aug. 2017.

[25] Ganache. [Online]. Available: https://www.trufflesuite.com/ganache

[26] Ahmed Kosba, A Java library for zk-SNARK circuits, 2015. [Online]. Available: https://github.com/akosba/jsnark

[27] SCIPR Lab, C++ library for zkSNARKs, 2014. [Online]. Available: https://github.com/scipr-lab/libsnark

[28] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, and M. Virza, "SNARKs for C: Verifying program executions succinctly and in zero knowledge," in *Proc. Annu. Cryptol. Conf.*, 2013, pp. 90–108.

[29] ether camp, Java implementation of the Ethereum yellow paper, 2016. [Online]. Available: github.com/maxilbert/ethereumj

[30] M. Raya and J.-P. Hubaux, "Securing vehicular ad hoc networks," *J. Comput. Secur.*, vol. 15, no. 1, pp. 39–68, 2007.

[31] Solidity, the Smart Contract Programming Language, 2007. [Online]. Available: https://github.com/ethereum/solidity

[32] ETH Gas Station: Consumer oriented metrics for the Ethereum gas market, *Etherum Gas Station*, [Online]. Available: https://ethgasstation.info/

[33] M. Li, D. Hu, C. Lal, M. Conti, and Z. Zhang, "Blockchain-enabled secure energy trading with verifiable fairness in industrial internet of things," *IEEE Trans. Ind. Informat.*, vol. 16, no. 10, pp. 6564–6574, Oct. 2020.

[34] V. Hassija, V. Chamola, S. Garg, N. G. K. Dara, G. Kaddoum, and D. N. K. Jayakody, "A blockchain-based framework for lightweight data sharing and energy trading in V2G network," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 5799–5812, Jun. 2020.

[35] G. Kumar *et al.*, "A privacy-preserving secure framework for electric vehicles in IoT using matching market and signcryption," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7707–7722, Jul. 2020.

[36] N. Z. Aitzhan and D. Svetinovic, "Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 5, pp. 840–852, Sep./Oct. 2018.

Mohamed Baza received the B.S. and M.S. degrees in electrical and computer engineering from Benha University, Banha, Egypt, in 2012 and 2017, respectively, and the Ph.D. degree in electrical and computer engineering from Tennessee Tech University, Cookeville, TN, USA, in December 2020. He is currently an Assistant Professor with the Department of Computer Science, College of Charleston, Charleston, SC, USA. From August 2020 to May 2021, he was a Visiting Assistant Professor with the Department of Computer Science, Sam Houston State University, Huntsville, TX, USA. He also has more than two years of industry experience in information security in Apache-khalda petroleum company, Egypt. He is the author of numerous papers published in major IEEE conferences and journals, including the IEEE Wireless Communications and Networking Conference, IEEE International Conference on Communications, IEEE Vehicular Technology Conference, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS OF VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING, and IEEE SYSTEMS JOURNAL. His research interests include blockchains, cyber-security, machine learning, smart-grid, and vehicular ad-hoc networks. He was a Reviewer for various journals and conferences including the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE INTERNET OF THINGS JOURNAL, and journal of *Peer-to-Peer Networking*. He was the recipient of the Best IEEE Paper Award in the International Conference on Smart Applications, Communications, and Networking (SmartNets 2020).

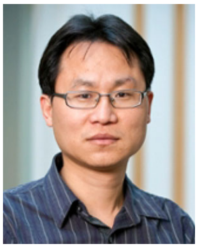
Ahmed Sherif received the M.Sc. degree in computer science and engineering from the Egypt-Japan University of Science and Technology, New Borg El Arab, Egypt, in 2014 and the Ph.D. degree in electrical and computer engineering from Tennessee Tech University, Cookeville, TN, USA, in August 2017. He is currently an Assistant Professor with the School of Computing Sciences and Computer Engineering, The University of Southern Mississippi, Hattiesburg, MS, USA. He is the author of numerous papers published in major IEEE conferences and journals, including the IEEE International Conference on Communications, IEEE Vehicular Technology Conference, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, and IEEE INTERNET OF THINGS JOURNAL. His research interests include cybersecurity, security, and privacy-preserving schemes in autonomous vehicles, vehicular ad hoc networks, Internet of Things applications, and smart grid advanced metering infrastructure network. He was a Reviewer for various journals and conferences, including the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE INTERNET OF THINGS JOURNAL, and the journal of *Peer-to-Peer Networking*.

Mohamed M. E. A. Mahmoud is currently an Associate Professor with the Department of Electrical and Computer Engineering, Tennessee Tech University, Cookeville, TN, USA. From May 2011 to May 2012, he was a Postdoctoral Fellow with the Broadband Communications Research Group, University of Waterloo, Waterloo, ON, USA. From August 2012 to July 2013, he was a Visiting Scholar with the University of Waterloo, and a Postdoctoral Fellow with Ryerson University, Toronto, ON, Canada. He is the author for more than twenty three papers published in major IEEE conferences and journals, including the INFOCOM conference and IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, *Mobile Computing*, and *Parallel and Distributed Systems*. His research interests include security and privacy preserving schemes for smart grid communication network, mobile ad hoc network, sensor network, and delay-tolerant network. He was the recipient of the NSERC-PDF Award. He won the Best Paper Award from IEEE International Conference on Communications (ICC’09), Dresden, Germany, 2009. He is an Associate Editor for the Springer journal of *Peer-to-Peer Networking and Applications*. He was a Technical Program Committee Member of various IEEE conferences and a Reviewer of various journals and conferences, including the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, and the journal of *Peer-to-Peer Networking*.

Spiridon Bakiras received the B.S. degree in electrical and computer engineering from the National Technical University of Athens, Athens, Greece, in 1993, the M.S. degree in telematics from the University of Surrey, Guildford, U.K., in 1994, and the Ph.D. degree in electrical engineering from the University of Southern California, Los Angeles, CS, USA, in 2000. He is currently an Associate Professor with the College of Science and Engineering, Hamad Bin Khalifa University, Al Rayyan, Qatar. Before that, he held teaching and research positions with Michigan Technological University, City University of New York, New York City, NY, USA, The University of Hong Kong, Hong Kong, and Hong Kong University of Science and Technology, Hong Kong. His current research interests include database security and privacy, mobile computing, and spatiotemporal databases. He is a Member of the ACM, and was the recipient of the U.S. National Science Foundation CAREER award.

Waleed Alasmary (Senior Member, IEEE) received the B.Sc. degree (Hons.) in computer engineering from Umm Al-Qura University, Mecca, Saudi Arabia, in 2005, the M.A.Sc. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2010, and the Ph.D. degree in electrical and computer engineering from the University of Toronto, Toronto, ON, Canada, in 2015. During the Ph.D. degree, he was a Visiting Research Scholar with Network Research Laboratory, University of California, Los Angeles, Los Angeles, CA, USA, in 2014. From 2016 to 2017, he was a Fulbright Visiting Scholar with CSAIL Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA. He subsequently joined the College of Computer and Information Systems, Umm Al-Qura University, as an Assistant Professor of computer engineering, where he currently holds an Associate Professor position. His current research interests include mobile computing, ubiquitous sensing, and intelligent transportation systems, privacy, and anonymity. His Mobility impact on the IEEE 802.11p article is among the most cited Ad Hoc Networks journal articles list from 2016–2018.

Mohamed Abdallah received the B.Sc. degree from Cairo University, Cairo, Egypt, in 1996, and the M.Sc. and Ph.D. degrees from the University of Maryland, College Park, MD, USA, in 2001 and 2006, respectively. From 2006 to 2016, he held academic and research positions with Cairo University, Giza, Egypt and Texas AM University at Qatar, Al Rayyan, Qatar. He is currently a Founding Faculty Member with the rank of Associate Professor with the College of Science and Engineering, Hamad bin Khalifa University, Ar-Rayyan, Qatar. He has authored or coauthored more than 150 journals and conferences and four book chapters, and co-invented four patents. His current research interests include wireless networks, wireless security, smart grids, optical wireless communication, and blockchain applications for emerging networks. He was the recipient of the Research Fellow Excellence Award at Texas AM University at Qatar in 2016, the Best Paper Award in multiple IEEE conferences, including the IEEE BlackSeaCom 2019 and the IEEE First Workshop on Smart Grid and Renewable Energy in 2015, and the Nortel Networks Industrial Fellowship for five consecutive years, 1999–2003. His professional activities include an Associate Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS and IEEE OPEN ACCESS JOURNAL OF COMMUNICATIONS, the Track Co-Chair of the IEEE VTC Fall 2019 conference, the Technical Program Chair of the 10th International Conference on Cognitive Radio Oriented Wireless Networks, and a Technical Program Committee Member of various major IEEE conferences.



Xiaodong Lin (Fellow, IEEE) received the Ph.D. degree in Information Engineering from Beijing University of Posts and Telecommunications, China, and the Ph.D. degree (with Outstanding Achievement in Graduate Studies Award) in Electrical and Computer Engineering from the University of Waterloo, Canada. He is currently a full Professor in the School of Computer Science, University of Guelph, Canada. His research interests include wireless communications and network security, computer forensics, privacy-enhancing technologies, Blockchain and applied cryptography. He is a Fellow of the IEEE.