# Blockchain-Based Credential Management for Anonymous Authentication in SAGVN

Dongxiao Liu, *Member, IEEE*, Huaqing Wu, *Member, IEEE*, Cheng Huang, *Member, IEEE*, Jianbing Ni, *Senior Member, IEEE*, and Xuemin Shen, *Fellow, IEEE*

***Abstract*— In this paper, we propose a blockchain-based collaborative credential management scheme for anonymous authentication in space-air-ground integrated vehicular networks (SAGVN), named *SAG-BC*. First, we build a consortium blockchain among service providers and design a distributed system setup (DSS) scheme to securely generate public parameters for issuing credentials. Second, we design a collaborative credential issuance (CCI) scheme to generate a succinct and easy-to-manage subscription credential. The credential can be used by users to access different access points in SAGVN efficiently without revealing true identities from the authentication messages. With co-designs of zero-knowledge proofs and succinct on-chain commitments, *SAG-BC* provides efficient verifiability and incentives for credential management operations in SAGVN. By doing so, expensive on-chain storage and computational overheads are reduced in the DSS and CCI. Finally, we conduct a thorough security analysis to demonstrate that *SAG-BC* achieves security and verifiability for credential management in SAGVN. We set up a real-world blockchain network and conduct extensive experiments to show the feasibility and efficiency of *SAG-BC*.**

***Index Terms*— Blockchain, anonymous authentication, credential management, space-air-ground integrated vehicular networks (SAGVN).**

## I. Introduction

SATELLITES and unmanned aerial vehicles (UAVs) can be integrated with the terrestrial vehicular networks, forming a space-air-ground integrated vehicular network (SAGVN), to support anywhere and anytime vehicular communications and services [1]. To further improve the resource utilization efficiency and enhance communication quality, antenna array-enabled beamforming technologies [2] can be deployed in SAGVN. By concentrating the transmitted energy in desired directions, flexible beamforming [3], [4] and cooperative transmissions by base stations, UAVs, etc. [5] can be achieved to enhance vehicular applications with ubiquitous coverage, reliable connections, and satisfied quality of service (QoS) [1]. At the same time, due to the highly directional communication beams and high vehicle/UAV/satellite mobilities, frequent misalignment of beams may happen, leading to radio link failures and thus exacerbating the frequent handover issues [6].

In SAGVN, authentication mechanism is essential for vehicular users (VEs) and space/air/ground access points (APs) to authenticate each other and establish secure wireless communications. As the frequent handovers in SAGVN result in frequent executions of authentication protocols, designing an efficient authentication protocol for antenna array-enabled SAGVN raises various challenges: First, exchanged messages between users and APs in the authentication can contain sensitive user identity information. Such information can be intercepted by an external attacker in the wireless channel [7], [8] to infer user's location, trace, and even daily routines. Second, the frequent handovers caused by the dense deployment of heterogeneous APs and the beamforming misalignment also increase the credential management complexity. A deployed solution in the mobile network is to let the serving AP contact the home network (HN) of users for subscription validations, which may increase the authentication delay especially for satellite APs [9] in SAGVN. Alternatively, users can obtain different credentials to directly access different APs. Such a mechanism may increase the credential management burden on users since the users may need to contact different service providers to obtain service credentials.

Generally, there is usually a centralized and trusted credential manager [7], [10], [11] to authenticate users and assign subscription credentials, which may not be reliable for heterogeneous service providers in SAGVN due to the following issues: (1) The centralized credential manager increases the risk of single point failure [12]. For the heterogeneous network access, decentralized identity management mechanisms can be a promising solution to increase the system robustness and security; (2) Since the user credential is used to access network resources, it can be difficult for service providers from different domains to agree on a single authority to manage service subscriptions [13], [14]. To address the issues, blockchain can serve as a distributed database and the smart contract technique is used to manage user credentials with the blockchain. First, the integration of the blockchain can enhance the security of the distributed credential management.

Manuscript received 30 September 2021; revised 21 January 2022; accepted 25 February 2022. Date of publication 10 August 2022; date of current version 16 September 2022. *(Corresponding author: Jianbing Ni.)*

Dongxiao Liu, Cheng Huang, and Xuemin Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: dongxiao.liu@uwaterloo.ca; cheng.huang@uwaterloo.ca; sshen@uwaterloo.ca).

Huaqing Wu is with the Department of Electrical and Software Engineering, University of Calgary, Calgary, AB T2N 1N4, Canada (e-mail: huaqing.wu1@ucalgary.ca).

Jianbing Ni is with the Department of Electrical and Computer Engineering, Queen's University, Kingston, ON K7L 3N6, Canada (e-mail: jianbing.ni@queensu.ca).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSAC.2022.3196091.

Digital Object Identifier 10.1109/JSAC.2022.3196091

Specifically, the blockchain can be a secure broadcast channel [28] to ensure reliable message deliveries and a consistent global view of the credential management among all participants. Without the channel, an adversary that controls the network can utilize the asynchronization of messages and exploit the inconsistent view between participants to make insecure credential generation queries in SAGVN. Second, since insufficient credential management transparency [15] was provided for service providers, blockchain can serve as an immutable ledger and programmable environment to faithfully record related actions in the distributed credential management, which are secure and reliable provenance and forensics data [16] to help enforce regulatory operations.

While there are studies on blockchain-based credential management in mobile/air/space networks [12]–[14], [17], they cannot be efficiently applied to SAGVN due to the unique requirements of a subscription credential as mentioned above. Moreover, since on-chain resources (computing and storage) are expensive and open to blockchain participants, an everything-on-chain strategy can increase the on-chain overheads and the risk of credential leakage. Therefore, it is challenging to conduct the credential management efficiently and securely on the blockchain.

In this paper, we propose a blockchain-based credential management scheme, named *SAG-BC*, for anonymous authentication in SAGVN. The main contributions of this paper are summarized as follows:

- We design a blockchain-based distributed system setup (DSS) to enable service providers to generate public parameters and key shares for the credential management in SAGVN. Based on the DSS, we design a blockchain-based collaborative credential issuance (CCI), where service providers can generate partial credentials for user subscriptions in SAGVN. The partial credentials can be aggregated into a succinct credential, which can be used by users to access individual subscribed service without revealing their true identities from authentication messages.
- We design succinct commitments with zero-knowledge proofs for verifiable operations of participants in DSS and CCI. By utilizing off-chain communication channels with on-chain incentives and operation verifiability, the DSS and CCI achieve on-chain storage and computation efficiency for credential management in SAGVN.
- We demonstrate that *SAG-BC* achieves desired security requirements, including secure distributed system setup and credential issuance, credential security, and verifiability. We conduct extensive experiments on a real-world consortium blockchain to demonstrate the application feasibility and efficiency of *SAG-BC*.

The remainder of this paper is organized as follows. In Section II, we summarize the related works. We present the system model, security model, and design goals in Section III. In Section IV, we discuss the preliminaries of *SAG-BC*. The detailed constructions of *SAG-BC* are shown in Section V. We discuss the security properties in Section VI and the performance evaluation of *SAG-BC* in Section VII. Finally, we conclude this paper in Section VIII.

## II. Related Work

In this section, we discuss the related works of *SAG-BC*, including authentication mechanisms in SAGVN and the blockchain-based credential management.

### A. Authentication Mechanisms in SAGVN

Many research works were proposed for the credential management in vehicular networks [10], space information networks [9], UAV networks [12], and SAG integrated networks (SAGIN) [19]. In Universal Subscriber Identity Module (USIM)-based mobile networks, users are assigned with unique symmetric-key credentials [18]. Users authenticate themselves at APs with the help of HNs. Physical layer information can be utilized [20], [21] for key management and authentication in multiple wireless channels. In SAGVN, the frequent handovers and diversified service subscriptions can result in additional authentication delay if HN is highly involved in the authentication. The impact can be more significant in space networks due to the long communication delay between satellites and ground HNs.

For anonymous authentications, exiting standardizations for vehicular communications [10], [18], [22], [23] proposed to adopt pseudo identity-based mechanisms. A security credential management system for vehicular networks was proposed in [10] with careful discussion about the management cost when vehicles need to frequently change pseudonyms. Anonymous credentials based on group signature can be constructed [7] to save the management cost of user pseudo identities. Light-weight access authentication was proposed in [9] for space networks, where a network management center is utilized to authenticate roaming users. Identity-based mutual authentication protocol was proposed in [19] for SAGIN with the re-authentication mechanism to increase the network access efficiency. While existing works achieve rich functionalities for anonymous authentication, how to efficiently embed service subscription information into anonymous credentials still requires design considerations in SAGVN.

An efficient and anonymous authentication protocol for SAGVN was proposed in [11], that enables (1) directly mutual authentications between users and APs without consulting HNs, (2) anonymous service access between users and APs with early application data transmitted to reduce communication delays, and (3) efficient credential management with multi subscriptions into a succinct credential. Unlike existing works based on a single trusted credential manager, *SAG-BC* further investigates the distributed system setup and collaborative credential issuance of a multi-subscription credential and designs a consortium blockchain-based transparent and verifiable credential management mechanism specifically for SAGVN.

### B. Blockchain-Based Credential Management

Blockchain-based credential management schemes were proposed for vehicular networks [13], [14], where the blockchain serves as a distributed storage, and a trusted entity is involved to set up the system or manage the credential. Blockchain-based group key agreement [12] utilized the smart

contract to increase the management transparency in wireless networks. At the same time, smart contract techniques can also enhance liveness and robustness of the distributed key generation [24], [25]. Threshold cryptographic techniques were studied from traditional verifiable secret sharing (*VSS*) [26], [27] for scalable distributed key generation [28], [29]. *VSS* enables a dealer to securely share a secret with multiple parties in a verifiable manner [26], which can also be made publicly verifiable by encrypting the shares and proving correct generation with zero-knowledge proof technique [30]. In [31], *VSS* was integrated with anonymous credential techniques to achieve threshold identity credential issuance and revocation. Attribute credentials were also designed to achieve selective credential disclosure on the blockchain [32]. In SAGVN, the proposed identity or attribute credentials may not be efficiently adopted due to the unique requirements of subscription credentials as mentioned before.

To enhance the credential management security and promote collaborations among service providers, *SAG-BC* proposes a blockchain-based credential management scheme in SAGVN. Moreover, *SAG-BC* addresses the on-chain efficiency and verifiability issue for credential management in SAGVN by co-designing zero-knowledge proofs and succinct on-chain commitments.

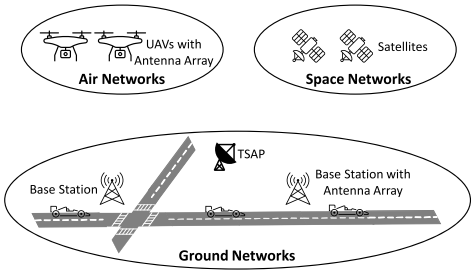



Fig. 1. System model of SAGVN.

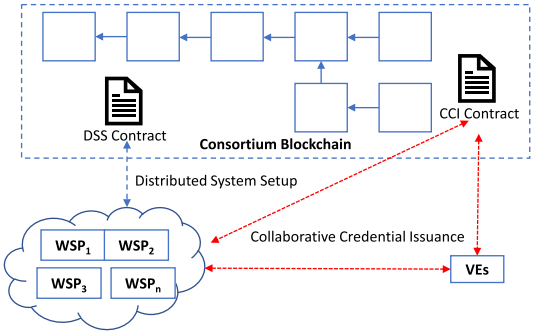



Fig. 2. Workflow of credential management.

## III. Problem Formulation

In this section, we formulate the system model and security model of *SAG-BC*, and present design goals.

### A. System Model

We consider a subscription-based service paradigm in SAGVN, where VEs can first subscribe to different services from various service providers and then access corresponding APs. VEs should obtain valid credentials that can prove their valid identity and service subscription when establishing secure connections with different APs. Specifically, there are four entities in *SAG-BC*:

- Access Point (AP): There are multiple APs enabled in SAGVN, including satellites, UAVs, base stations, and drive-through WiFis [33]. The heterogeneous APs can provide differentiated access services in SAGVN, from traditional voice/data services to emerging content catching services at the edge for autonomous vehicles [1].
- Wireless Service Provider (WSP): WSPs manage different APs and provide access services in SAGVN. WSPs can be mobile operators or satellite network operators. WSPs can form a consortium to collaboratively manage user subscriptions for authenticated network access and service charging.
- Blockchain (BC): BC is a shared ledger by WSPs, that consists of immutable and trustworthy management records [34] between WSPs. BC is critical for WSPs to collaboratively set up public parameters and enforce user credential issuance.
- Vehicular User (VE): VEs are moving objects in SAGVN requiring diversified services from different WSPs. They can be equipped with advanced communication transceivers to access heterogeneous APs.

In Figure 1, we describe a system model of antenna array-enabled SAGVN. We consider air networks of UAVs, space networks of satellites, and ground networks of base stations. Antenna arrays can be deployed at air and ground networks to increase spectrum efficiency and enable cooperative transmissions [5]. A terrestrial satellite access point (TSAP) can be deployed as a relaying node between VEs and satellite APs. A vehicle moving along the road can switch to different networks according to available network resources and its service requirements [1]. In SAGVN, VEs can conduct an efficient and anonymous authentication protocol [11] with APs using a multi-subscription credential.

In Figure 2, *SAG-BC* works with the following steps to generate a subscription credential for anonymous authentications: (1) WSPs form a consortium to set up a secure blockchain network, e.g., Hyperledger Fabric [35]. Each WSP can obtain a valid blockchain membership to access and manage the shared ledger. WSPs also set up two smart contracts: DSS and CCI; (2) WSPs interact with the DSS contract to securely generate public parameters of *SAG-BC* and register themselves with public service information; (3) VEs communicate with WSPs to subscribe to different services. VEs and WSPs interact with the CCI contract to generate an anonymous subscription credential. Note that, *SAG-BC* combines on-chain and off-chain communications to ensure that critical management operations are auditable on the blockchain.

### B. Security Model

VEs are rational users who may intend to free-ride access services in SAGVN. WSPs are rational entities that are

regulated by governmental offices. That is, they follow the designed DSS and CCI protocols since their activities in DSS and CCI are regulated and verifiable. BC is a secure ledger with immutable storage and verifiable state updates.

### C. Design Goals

Under the system model and security model, *SAG-BC* should achieve the following design goals:

- Security: (1) Public parameters for credential issuance are initialized distributively and securely by all WSPs; (2) A valid credential that embeds multiple service subscriptions is issued collaboratively by WSPs. Valid credentials cannot be issued without the help of at least $t$ WSPs; (3) Without the knowledge of the secret key of a valid credential, VEs cannot prove valid service subscriptions from the credential.
- Verifiability: In the distributed system setup and collaborative credential issuance process, VE and WSP operations should be verifiable.
- Efficiency: Service credential should be succinct regardless of the number of subscribed services. DSS and CCI contracts should be practical and feasible for real-world implementations.

## IV. Preliminaries

### A. Cryptographic Background

We consider pairing-based cryptography over elliptic curves. Specifically, a set of groups is denoted as $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ with a prime order $p$ and a Type III asymmetric bilinear pairing $e$. Let $\mathbb{Z}_p$ denote a ring of integers with the same order $p$. $[n]$ denotes a set of integers from 1 to $n$. $g, \tilde{g}$ denote generators from $\mathbb{G}_1, \mathbb{G}_2$, respectively. $1_{\mathbb{G}_1}, 1_{\mathbb{G}_2}$ denote identity generators from $\mathbb{G}_1, \mathbb{G}_2$, respectively.

The security of the pairing-based cryptography is based on one-way functionality over elliptic curves [36]. Specifically, given two random numbers $a, b \in \mathbb{Z}_p^2$, it is easy to calculate $g^a, \tilde{g}^b$ and $e(g^a, \tilde{g}^b)$, but the reversible computations cannot be calculated efficiently. Pairing-based cryptography has found many interesting applications, such as the digital signature and the anonymous credential [37].

Zero-knowledge proof (ZKP) in the discrete logarithm plays an important role. Informally, *ZKP* lets a prover with secrets in $\mathbb{Z}_p$ generate a proof $\pi$ to demonstrate a verifier that the secrets satisfy a public relation without leaking the secrets. A ZKP protocol can be written as [38]:

$$ZKP\{(a) : Y_1 = g^a \wedge \tilde{Y}_2 = \tilde{g}^a\}, \tag{1}$$

where $a$ is the secret and $g, \tilde{g}, Y_1, \tilde{Y}_2$ are public parameters. The above *ZKP* demonstrates the equality of $a$ over different generators, which can be utilized in *SAG-BC*. It can be instantiated via a non-interactive protocol [38] with two algorithms: (1) $Prove$ algorithm takes the public parameters and the secret to generate a proof $\pi$; (2) $Verify$ algorithm takes the public parameters and $\pi$ to output accept or reject. *ZKP* requires that a computationally-bounded prover cannot forge a proof (*soundness*), an honest verifier accepts the valid proof (*completeness*), and the verifier cannot learn the specific secret *(zero knowledge)*.

### B. PS Signature

PS signature [39] is a digital signature scheme from a Type III pairing: $(\mathbb{G}, \mathbb{Z}_p, e, g, \tilde{g})$. It sets $(x, y_1, y_2, \ldots, y_n) \in \mathbb{Z}_p^{n+1}$ as secret keys and $\tilde{X} = \tilde{g}^x, \tilde{Y}_1 = \tilde{g}^{y_1}, \ldots, \tilde{Y}_n = \tilde{g}^{y_n}$ as public keys. Given a set of messages $\mathcal{I} = \{m_i\} \in \mathbb{Z}_p^n$, a PS signature is computed as follows:

$$(\pi_1, \pi_2) = (g_r, g_r^{x+\sum_{i=1}^{n} m_i y_i}), \tag{2}$$

where $g_r \in \mathbb{G}_1$ is a random generator. A verifier can use the following equation to check the signature:

$$e(\pi_1, \tilde{X} \prod_{i=1}^{n} \tilde{Y}_i^{m_i}) = e(\pi_2, \tilde{g}). \tag{3}$$

Public keys can be generated by different entities with a proof of knowledge of the corresponding secret key. As a result, PS signature supports the aggregation of multi signatures under different keys [31]. Given $\pi_1, \pi_2$ on messages $\mathcal{I}$, it supports derivation of the original signature to generate a redacted signature $\pi_1', \pi_2', \tilde{\pi}_1', \tilde{\pi}_2'$ on a subset of messages, $\mathcal{S}_A \in \mathcal{I}$. The obtained redacted signature can be verified with corresponding public keys for $\mathcal{S}_A$ [40], [41]. The security of redatable PS signature has two folds: (1) *Unforgeability* ensures that an efficient adversary cannot forge a valid signature without knowing the secret keys; (2) *Unlinkability* ensures that the redacted signature does not leak information of messages $\mathcal{I} \setminus \mathcal{S}_A$.

The PS signature also supports randomization, such that the prover can raise the signature to the power of a randomness $r$. By combining it with the ZKP technique, a prover holding a valid signature can anonymously prove the knowledge of any single or subset of the signed messages $\mathcal{I}$.

### C. Threshold Cryptographic System

Threshold cryptographic system (*TCS*) is widely used in constructing distributed security systems, such as distributed signature and encryption [24], [42]. Essentially, *TCS* can break a secret $s$ into $n$ shares and any $t$-out-of-$n$ shares can recover the secret. For Shamir's secret sharing scheme [43], a dealer holding the secret $s$ can build a $(t-1)$-degree polynomial $P(x)$ as follows:

$$P(x) = s + \sum_{i=1}^{t-1} a_i x^i, \tag{4}$$

where $a_i \in \mathbb{Z}_p$ is the polynomial coefficient. To compute $n$ shares, the dealer evaluates $P(x)$ and gets $(P(1), P(2), \ldots, P(n))$. Suppose we have $t$ shares indexed by a set of integers $\mathcal{T} \in [n]$. For each $i \in \mathcal{T}$, we can calculate a Lagrange coefficient $\lambda_i = \prod_{j \in \mathcal{T}, j \neq i} \frac{j}{j-i}$. Then, the secret can be recovered as follows:

$$s = \sum_{i \in \mathcal{T}} P(i) \lambda_i. \tag{5}$$

*TCS* achieves the following properties (1) *Correctness*: Any valid $t$ shares can recover the same secret; (2) *Security*: The secret should be random and any less than $t$ shares cannot recover the secret.

To deal with a dishonest dealer, verifiable secret sharing (*VSS*) is introduced [26]. In *VSS*, a dealer computes shares and cryptographic commitments of the polynomial coefficients $(s, a_1, \ldots, a_{t-1})$. Any party with a share and the commitments can verify if the share is actually an evaluation of $P(x)$. For distributed *VSS*, the role of the dealer is broken into $n$ parties [28], [44]. Each party $p_i$ chooses a local secret $s_i$ and runs *VSS* with a local polynomial $P_i(x)$ to share the secret to $n$ shares. After receiving all shares, each party can verify and combine the shares into an aggregated share. VSS can also be made publicly verifiable [30] by encrypting the secret from $\mathbb{Z}_p$ with an encryption scheme.

### D. Consortium Blockchain

Consortium blockchain is a permissioned and distributed ledger. (1) *Permissioned* means only authenticated nodes can access the ledger. Therefore, compared with the public blockchain, the consortium blockchain implements more efficient consensus protocols with higher transaction throughput and less transaction confirmation time, e.g., Practical Byzantine Fault Tolerance (PBFT) or RAFT; (2) *Distributed* means that the consortium blockchain helps consortium members (i.e., WSPs in *SAG-BC*) to maintain a shared database with consistent storage *(Persistence)* and verifiable status updates *(Liveness)*. The consortium blockchain is a promising solution to inter-organizational business process [34], by improving management efficiency and promoting trustworthy collaboration. To enable flexible management of the ledger storage, smart contract as a distributed computer program is stored on the ledger and is executed by consortium members to update the ledger status.

## V. Constructions of SAG-BC

In this section, we first present an overview of *SAG-BC*. Then, we present detailed constructions of DSS and CCI. Notations are shown in Table I.

### A. Overview

In *SAG-BC*, VEs can subscribe to diversified services and obtain a subscription credential as a multi-message PS signature:

$$(h, h^{x+m'y_0+sk_v \sum_{AP_{i,j} \in \mathcal{S}_A} y_{i,j}}). \tag{6}$$

$h$ is a random generator; $sk_v$ is the secret key of a VE; $\mathcal{S}_A$ is a set of subscribed APs; $y_{i,j}$ is the public key of a subscribed $AP_{i,j} \in \mathcal{S}_A$; Accordingly, the public keys for the credential should be $(\tilde{X}, \tilde{Y}_0, \{\tilde{Y}_{i,j}\})$, where $\tilde{Y}_{i,j}$ corresponds to the $j_{th}$ AP of $w_i$. Compared with the original form of the multi-message PS signature, we focus on service subscriptions and simplify the original PS signature in Equ. 2. More specifically, *SAG-BC* sets $m_i = sk_v$ for subscribed APs and sets others as 0 [41]. If a secret key of $AP_{i,j}$ is included in the signature, it means a valid subscription in the credential for $AP_{i,j}$. We also add a random message $m'$ in the signature, which is for the security considerations of the original PS signature [31]. To generate the above credential, service providers work together to conduct distributed system setup (DSS) and collaborative credential issuance (CCI).

TABLE I
Notations

| | |
|---|---|
| $\lambda$ | Security parameter |
| $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ | Asymmetric elliptic groups |
| $e : \mathbb{G}_1 * \mathbb{G}_2 \to \mathbb{G}_T$ | Bilinear pairing |
| $g, \tilde{g}$ | Generators from $\mathbb{G}_1, \mathbb{G}_2$ |
| $p$ | Prime order |
| $\mathbb{Z}_p$ | Integers |
| $n$ | Number of WSPs |
| $t$ | Threshold number |
| $P(x)$ | $(t-1)$-degree polynomial |
| $[n]$ | $1, 2, ..., n$ |
| $(s, a_1, ..., a_{t-1})$ | Coefficients of $P(x)$ |
| $\mathcal{I}$ | Set of all APs |
| $\mathcal{S}_A$ | Set of subscribed APs |
| $w_i$ | Wireless service provider |
| $AP$ | Access point |
| $\mathcal{I}_{w_i}$ | Set of APs of $w_i$ |
| $H_0 : (0,1)^* \to \mathbb{Z}_p$ | Hash function $H_0$ |
| $H_1 : (0,1)^* \to \mathbb{G}_1 * \mathbb{Z}_p$ | Hash function $H_1$ |
| $\tilde{\mathcal{C}}$ | Set of Generators |
| $\pi$ | A ZKP proof |

In DSS, WSPs work together to generate private/public key pairs $(x, \tilde{X} = \tilde{g}^x)$, $(y_0, \tilde{Y}_0 = \tilde{g}^{y_0})$ and $\{y_{i,j}, \tilde{Y}_{i,j} = \tilde{g}^{y_{i,j}}\}$. $(x, y_0)$ are global secret keys and $\{y_{i,j}\}$ are secret keys for each access points. $(\tilde{X}, \tilde{Y}_0, \{\tilde{Y}_{i,j}\})$ are corresponding public keys. In the end of DSS, all secret keys should be securely shared among service providers and the public keys should be published on the blockchain. To reduce on-chain overheads, *SAG-BC* carefully tailors the polynomial commitments [44], [47] and threshold crypto systems [24], [25] to design an efficient blockchain-based credential management in SAGVN. To enhance the on-chain communication security, a commit-then-reveal phase is introduced to make the original polynomial commitments secure on the blockchain. With succinct commitments and zero-knowledge proofs for DSS, on-chain operations of service providers are efficiently verifiable to motivate service providers faithfully fulfill the DSS process.

In CCI, VEs first subscribe to various services and pay subscription fees to WSPs. Then, VEs communicate with the blockchain and WSPs to generate a subscription credential. To let different WSPs agree on the same random generator $h$ and the same random message $m'$, we use a hash function $H_1$ as a random oracle to map a public but unique string bounded to a VE to two group elements [31], [32]. Since *SAG-BC* is used for service subscriptions, we do not require blind signing on user-chosen secrets as in [32]. Instead, a user chooses a unique secret and publishes two computationally hiding commitments on the blockchain, which are also checked for their global uniqueness. More importantly, we fully combine the on-chain and off-chain channels with incentive mechanisms to make the CCI efficient, secure, and verifiable.

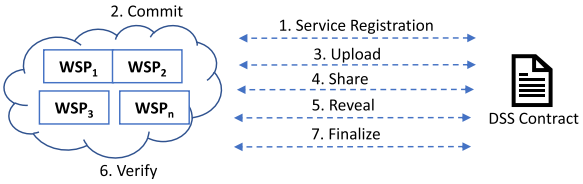



Fig. 3. Workflow of DSS.

For illustrative purposes, we make the following assumptions in *SAG-BC*:

- We consider a synchronous communication model with the combination of on-chain and off-chain channels: (1) BC is a secure and reliable on-chain channel; (2) Secure and authenticated off-chain channels exist between WSPs and VEs.
- There are $n$ WSPs in the system. All WSPs are verified to ensure they can provide eligible services in SAGVN. Secure and authenticated off-chain channels are available between WSPs.
- All WSPs agree on a signature scheme, such as ECDSA. Each WSP $w_i$ holds a secret signing key $sk_{w_i}$, while the corresponding verification key $pk_{w_i}$ is known to the public.
- A consortium blockchain $(BC)$ is set up by WSPs. Each WSP has an authenticated credential (e.g., Fabric membership certificate) to interact with the blockchain in a secure and authenticated manner. VEs communicate with the blockchain with an authentication token, such as a token from a (distributed) subscription server.
- The DSS and CCI contracts are approved by all WSPs and implemented over $BC$. The total number of WSPs $(n)$ and the threshold number $(t)$ are pre-determined by the wireless regulator. WSPs are indexed by a simple increasing sequence of numbers $[n]$. The contracts can check the uniqueness of all identifiers or group elements stored on the blockchain. The contracts can also check that group elements on the blockchain are non-identity elements.
- WSPs have run a secure protocol [45] to compute a set of generators, including $\tilde{\mathcal{C}} = \{\tilde{c}_i = \tilde{g}^{s_t^i}\}_{i\in[0,t-1]}$ and $g^{s_t}$. $s_t$ is a trapdoor secret and is not leaked to any WSPs.

In the following, we present the detailed constructions of DSS and CCI.

### B. Distributed System Setup

WSPs agree on a system security parameter $\lambda$ and elliptic groups $\mathbb{G}$ with a prime order $p$ and a Type III pairing $e$. WSPs agree on non-identity random generators $g \in \mathbb{G}_1$ and $\tilde{g} \in \mathbb{G}_2$. The curve parameters are well-determined and the well-formedness of the random generators can be guaranteed [45]. As shown in Figure 3, the distributed system setup consists of the following phases:

*Service Registration* – Each WSP first registers its services as a set of $n_{w_i}$ APs:

$$\mathcal{I}_{w_i} = (AP_{i,1}, AP_{i,2}, \ldots, AP_{i,n_{w_i}}), \tag{7}$$

at a wireless regulator to obtain a service license $L_{w_i}$. Each WSP $w_i$ also computes a public/private key pair $(\tilde{pk}_i, sk_i)$ for an encryption scheme [46]:

$$\begin{aligned} \tilde{pk}_i &= \tilde{g}^{sk_i}, \\ \pi_{w_i} &= ZKP\{(sk_i) : \tilde{pk}_i = \tilde{g}^{sk_i}\}. \end{aligned} \tag{8}$$

The WSP uploads $(\mathcal{I}_{w_i}, L_{w_i}, \tilde{pk}_i, \pi_{w_i})$ to the DSS contract. After receiving the key registration message, the contract verifies the correctness of $L_{w_i}$ and $\pi_{w_i}$ and stores them if the verification passes. Then, WSPs work together to generate key pairs $(x, \tilde{X})$, $(y_0, \tilde{Y}_0)$ and $\{y_{i,j}, \tilde{Y}_{i,j}\}$ for PS signature. For illustrative purposes, we only describe the generation of $(x, \tilde{X})$ in details, which includes commit, upload, share, reveal, verify, and finalize. Other parameters can be generated using the same procedure.

*Commit* – Each registered WSP can participate in generating a distributed secret $x$ and a corresponding public key $\tilde{X} = \tilde{g}^x$ for the PS signature. Specifically, each WSP $w_i$ chooses a local secret $s_{w_i}$ and a $(t-1)$-degree polynomial $P_{w_i}(x)$ with coefficients $(a_{i,0}, a_{i,1}, \ldots, a_{i,t-1})$, where $a_{i,0} = s_{w_i}$. $w_i$ commits to the polynomial $P_{w_i}(x)$ as follows:

$$\tilde{C}_{w_i} = \prod_{j=0}^{t-1} \tilde{c}_j^{a_{i,j}} = \tilde{g}^{P_{w_i}(s_t)}. \tag{9}$$

Note that, this is a succinct polynomial commitment from [47] and is computed by using $\tilde{\mathcal{C}} = \{\tilde{c}_i = \tilde{g}^{s_t^i}\}_{i\in[0,t-1]}$. $s_t$ is the trapdoor secret and is not known in the computation.

$w_i$ evaluates $P_{w_i}(x)$ at $j \in [0, n]$ to obtain $P_{w_i}(j)$. For each $P_{w_i}(j)$, $w_i$ computes the commitment of a target polynomial as follows:

$$\begin{aligned} T_{w_i,j}(x) &= (P_{w_i}(x) - P_{w_i}(j))/(x-j), \\ \tilde{C}_{w_i,j} &= \tilde{g}^{T_{w_i,j}(s_t)}. \end{aligned} \tag{10}$$

$T_{w_i,j}(x)$ is the target polynomial. $T_{w_i,j}(s_t)$ is the evaluation of $T_{w_i,j}(x)$ at $s_t$ and $\tilde{C}_{w_i,j}$ can be calculated with generators in $\tilde{\mathcal{C}}$.

To generate public commitments for each point $(j, P_{w_i}(j))$, $j \in [0, n]$, $w_i$ computes the following equations:

$$\begin{aligned} \tilde{C}'_{w_i,j} &= \tilde{g}^{P_{w_i}(j)}, \\ \pi_{w_i,j} &= ZKP\{(P_{w_i}(j)) : \tilde{C}'_{w_i,j} = \tilde{g}^{P_{w_i}(j)}\}. \end{aligned} \tag{11}$$

*Upload* – Each WSP $w_i$ encrypts the polynomial commitment $\tilde{C}_{w_i}$ using its public encryption key $\tilde{pk}_i$ as:

$$\begin{aligned} \tilde{E}_{w_i} &= (\tilde{E}_{w_i,1}, \tilde{E}_{w_i,2}) \\ &= (\tilde{g}^{r_i}, \tilde{pk}_i^{r_i} \tilde{C}_{w_i}), \end{aligned} \tag{12}$$

where $r_i$ is chosen randomly from $\mathbb{Z}_p$. The WSP $w_i$ uploads $\tilde{E}_{w_i}$ to the DSS contract. The contract checks the message sender and ensures that elements are unique and non-identity. The contract stores the valid commitments in the blockchain storage. After all WSPs store their commitments on the contract, the DSS moves to the next phase.

*Share* – Each WSP $w_i$ encrypts $(\tilde{C}_{w_i,j}, \tilde{C}'_{w_i,j}, \pi_{w_i,j}, P_{w_i}(j))$ using $w_j$'s public encryption key $\tilde{pk}_j$ [24], which can result in multiple ElGamal encryptions $\tilde{E}_{i,j}$ [49]

considering the length of the plaintext. For illustrative simplicity, there is a reversible function that can convert the plaintext into $\mathbb{G}_2$ and vice versa. $w_i$ sends $(w_j, \tilde{E}_{i,j})$ to the DSS contract on the blockchain.

The contract checks the message sender and receiver information. After all peer-to-peer shares are checked by the DSS contract and recorded on the blockchain, the DSS moves to the next phase.

*Reveal* – Each WSP $w_i$ reveals its polynomial commitment by decrypting $\tilde{E}_{w_i}$ and generating a decryption proof as follows:

$$\pi_{w_i,D} = ZKP\{(sk_i) : \tilde{E}_{w_i,1}^{sk_i} \wedge \tilde{pk}_i = \tilde{g}^{sk_i}.\} \tag{13}$$

$\pi_{w_i,D}$ is proof of correct decryption, which is also used in distributed ElGamal systems [50]. The WSP $w_i$ uploads $(\tilde{E}_{w_i,1}^{sk_i}, \pi_{w_i,D}, \tilde{C}_{w_i,0}, \tilde{C}'_{w_i,0}, \pi_{w_i,0})$ to the DSS contract. The contract checks the message sender is $w_i$ and the correctness of $\pi_{w_i,D}$ and $\pi_{w_i,0}$. If the checks pass, the contract computes $\tilde{C}_{w_i} = \tilde{E}_{w_i,2}/\tilde{E}_{w_i,1}^{sk_i}$ and checks the $\tilde{C}'_{w_i,0}$ as follows:

$$e(g, \tilde{C}_{w_i}/\tilde{C}'_{w_i,0}) = e(g^{s_t}/g^0, \tilde{C}_{w_i,0}), \tag{14}$$

$g^{s_t}$ is a public generator. After all WSPs reveal the commitments and the checks pass, the DSS moves to the next phase.

*Verify* – Each $w_j$ retrieves all $\tilde{C}_{w_i}$ and $\tilde{E}_{i,j}$ from the DSS contract, where $i \in [1,n] \setminus j$. $w_j$ decrypts $\tilde{E}_{i,j}$ using its secret key $sk_j$ to obtain $\{\tilde{C}_{w_i,j}, \tilde{C}'_{w_i,j}, \pi_{w_i,j}, P_{w_i}(j)\}_{i\in[1,n]\setminus j}$. Each $w_j$ checks $\pi_{w_i,j}$ and the following equation:

$$e(g, \tilde{C}_{w_i}/\tilde{C}'_{w_i,j}) = e(g^{s_t}/g^j, \tilde{C}_{w_i,j}). \tag{15}$$

The WSP $w_j$ also checks whether its own share is correctly computed: $\tilde{C}'_{w_i,j} = \tilde{g}^{P_{w_i}(j)}$. If all checks of all WSPs pass, the DSS moves to the next phase.

*Finalize* – Until this stage, *all* WSPs have correctly shared and verified all their shares. The contract computes the PS public key $\tilde{X}$ as follows:

$$\tilde{X} = \prod_{i\in[n]} \tilde{C}'_{w_i,0}. \tag{16}$$

For each $w_i$, it computes the local share of $x$ as follows:

$$x_i = \sum_{j\in[n]} P_{w_j}(i). \tag{17}$$

After the execution of the DSS contract, each WSP $w_k$ successfully registers its service and obtains local shares $(x_k, y_{0,k}, \{y_{i,j,k}\})$. With the local shares, WSPs can collaboratively issue subscription credentials to VEs.

*Verifiability of DSS* – The DSS provides verifiability of each WSP behavior. (1) All communications in DSS happen on the blockchain that are authenticated and secure; (2) The polynomial commitments are first encrypted and stored onto the blockchain. Later, due to the use of ZKP, the encryptions can only be correctly decrypted under a WSP's secret key; (3) The shares between WSPs are transmitted under each other's public key. In case of any dispute, a receiving WSP can prove the correct decryption of the encrypted shares. The decrypted communications can be further verified for valid polynomial evaluations; (4) The correctness of ZKPs and the evaluations of polynomial commitments at given positions are all guaranteed due to the security of the polynomial commitment and the ZKP technique.

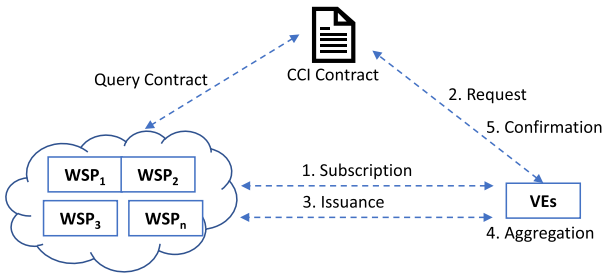



Fig. 4. Workflow of CCI.

*Discussions:* (1) Our DSS protocol is instantiated specifically for our model in SAGVN, where any WSP is motivated to complete the protocol and is not allowed to quit during the protocol. This is reasonable since WSPs would like to participate into the credential issuance process and their behavior in DSS is verifiable. Moreover, WSPs cannot afford to quit or to be disqualified from the DSS protocol since failing to behave correctly will lead to exclusion from providing services in SAGVN and potential liability enforcements; (2) Before the commitments of polynomials are revealed on the blockchain, no single WSP can guess the final outputs of public commitments and the associated secrets. As the DSS is completed with all WSPs, the randomness of the final outputs is achieved. Moreover, a protocol that re-generates the public keys on a new random base can also be considered [51]; (3) The share phase may be moved into secure and authenticated off-chain channels between WSPs. In case of message non-delivery, a message receipt can then require the non-delivered message to be encrypted and transmitted on the blockchain.

### C. Collaborative Credential Issuance

With the generated system public parameters and public keys for PS signature in the DSS, WSPs can work collaboratively to generate subscription credentials for VEs. As shown in Figure 4, the CCI consists of the following phases:

*Subscription* – A VE $v_i$ first obtains an authentication token from a (distributed) subscription server via a secure and authenticated channel. Specifically, the VE authenticates itself using real-world identities at the subscription server to get a unique pseudo ID $ID_{v_i}$ and a public signature key $pk_{v_i}$ with a corresponding signing key $sk_{v_i}$. The VE also chooses a secret $id_{v_i} \in \mathbb{Z}_p$. The VE computes:

$$\begin{aligned}(h_{v_i}, m') = H_1(ID_{v_i}||pk_{v_i}), T_v = h_{v_i}^{id_{v_i}}, T'_v = g^{id_{v_i}},\\ \pi_{v_i} = ZKP\{(id_{v_i}) : T_v = h_{v_i}^{id_{v_i}} \wedge T'_v = g^{id_{v_i}}\}.\end{aligned} \tag{18}$$

Based on service requirements and planned trips, the VE subscribes to a set of APs as $\mathcal{S}_A = \{AP_{i,j}\}$, where $AP_{i,j}$ is from a WSP $w_i$. Note that each $w_i$ can manage a set of APs, denoted as $\mathcal{I}_{w_i}$. The VE sends $T_v$, $T'_v$ and $\pi_{v_i}$ to the

subscription server via a secure and authenticated channel and pays the service fee for $\mathcal{S}_A$ via an off-chain payment channel.

The subscription server checks the message sender is the VE, verifies the correctness of $h_{v_i}$ and $\pi_{v_i}$, and checks the uniqueness of $h_{v_i}$, $T_v$, and $T'_v$. If the verification passes and the correct subscription fee is received, the subscription server generates an signed (by the server) token $Token_{v_i} = (ID_{v_i}, pk_{v_i}, T_v = h_{v_i}^{id_{v_i}}, T'_v = g^{id_{v_i}}, \pi_{v_i}, TS, H_0(S_A))$ and sends the token to the VE, where $TS$ is a current time stamp. The token can help WSPs to authenticate valid identity of $v_i$ and issue a service subscription credential following the form of Equ. 6. After obtaining the token, the VE checks the validity of the token and conducts CCI with WSPs on the blockchain.

*Credential Request* – The VE generates a request message:

$$m_t = (rid, Token_{v_i}, TS', Dep_{v_i}), \tag{19}$$

where $rid$ is a unique request ID, $TS'$ is a time stamp, and $Dep_{v_i}$ is the cryptocurrency deposit from the VE. The VE generates a signature on $m_t$ using the signing key $sk_{v_i}$ as $Sign_{sk_{v_i}}(m_t)$. The VE uploads $(m_t, Sign_{sk_{v_i}}(m_t))$ to the CCI contract. The contract checks the uniqueness of $rid$, $ID_{v_i}$, $pk_{v_i}$, $T_v$ and $T'_v$, and the message freshness. The contract checks that the token is signed by the subscription server. The contract verifies the correctness of $Sign_{sk_{v_i}}(m_t)$ by extracting $pk_{v_i}$ from $m_t$ and checks the $\pi_{v_i}$. The contract stores the valid $m_t$ on the blockchain and chooses a random set of at least $t$ WSPs as $\mathcal{S}_W$ for constructing a subscription credential. After seeing $S_W$ on the contract, for each WSP in $\mathcal{S}_W$, the VE sets:

$$m'_t = (m_t, \mathcal{S}_A, TS''), \tag{20}$$

where $TS''$ is a new time stamp, and sends $(m'_t, Sign_{sk_{v_i}}(m'_t))$ to the WSPs in $\mathcal{S}_W$ using a secure off-chain channel.

*Credential Issuance* – Upon receiving $m'_t$ from the VE $v_i$, the WSPs first extract $m_t$ from the CCI contract via $rid, ID_{v_i}$, and checks the consistency between $m_t$ and $m'_t$. Then, the WSPs check the correctness of $Sign_{sk_{v_i}}(m'_t)$ and $H_0(S_A)$ is consistent with the on-chain storage. The WSPs check that $h_{v_i}$, $T_v$ and $T'_v$ are unique and non-identity elements. The WSPs compute $(h_{v_i}, m') = H_1(ID_{v_i}||pk_{v_i})$ and check the correctness of $\pi_{v_i}$. If all checks pass, the WSPs generate a partial subscription token for $S_A$ as follows:

- For each WSP $w_k$ in $\mathcal{S}_W$, it computes:

$$Tok_{w_k} = h_{v_i}^{x_k} h_{v_i}^{m' y_{0,k}} T_v^{\sum^{AP_{i,j} \in \mathcal{S}_A} y_{i,j,k}}. \tag{21}$$

Note that $h_{v_i}$, $T_v$ and $T'_v$ are stored on the blockchain and are known to the VE and all the WSPs, which are also ensured to be globally unique. Each WSP sends its partial token $Tok_{w_k}$ and a signature using $sk_{w_k}$ as $Sign_{sk_{w_k}}(rid||ID_{v_i}||Tok_{w_k})$ to the VE via an off-chain secure channel.

*Credential Aggregation* – After receiving all partial tokens from the WSPs in $\mathcal{S}_W$, the VE can aggregate the tokens to obtain a subscription credential as follows:

$$\begin{aligned} Cerd_{v_i} &= \prod^{w_k \in \mathcal{S}_W} Tok_{w_k}^{\lambda_k} \\ &= \prod^{w_k \in \mathcal{S}_W} (h_{v_i}^{x_k + m' y_{0,k}} T_v^{\sum^{AP_{i,j} \in \mathcal{S}_A} y_{i,j,k}})^{\lambda_k} \\ &= h_{v_i}^{x + y_0 m'} h_{v_i}^{id_{v_i} \sum^{AP_{i,j} \in \mathcal{S}_A} y_{i,j}}. \end{aligned} \tag{22}$$

$\lambda_k = \prod_{w_j \in \mathcal{S}_W, j \neq k} \frac{j}{j-k}$. $Cerd_{v_i}$ is the PS signature on $id_{v_i}$ at locations of corresponding public keys of $AP_{i,j} \in S_A$.

*Credential Confirmation* – The VE can verify the correctness of $Cerd_{v_i}$ as follows:

$$e(h_{v_i}, \tilde{X}\tilde{Y}_0^{m'} \prod^{AP_{i,j} \in \mathcal{S}_A} \tilde{Y}_{i,j}^{id_{v_i}}) \stackrel{?}{=} e(Cerd_{v_i}, \tilde{g}). \tag{23}$$

If the verification passes, the VE generates a confirmation message as $m_c = (rid, ID_{v_i})$ and sends $(m_c, Sign_{sk_{v_i}}(m_c))$ to the CCI contract. The contract verifies the consistency between the confirmation and the original request via $rid, ID_{v_i}$ and checks the correctness of signatures using $pk_{v_i}$. This is to ensure the request and confirmation are sent from the same VE. If all checks pass, the deposit of the VE $Dep_{v_i}$ goes to WSPs in $S_W$.

*Verifiability of CCI* – During the above process, the VE and the WSPs can detect improper behavior of each other. In this case, the blockchain can serve as an immutable provenance ledger to process any dispute.

If the verification of the aggregated credential in Equ. 23 is not valid, one or more partial credentials received by the VE are not correct. In this case, WSPs can start a process to check the correctness of $Tok_{w_k}$. First, WSPs need to publish their individual public keys. More specifically, for each public key, WSPs reveal all the received $(\tilde{C}_{w_i,j}, \tilde{C}'_{w_i,j}, \pi_{w_i,j})$ in the share phase of the DSS onto the blockchain. The blockchain checks the correctness of the shares similarly to the reveal phase. Then, public keys for an individual WSP $w_j$ can be computed by aggregating the correct evaluations of polynomials $\tilde{C}'_{w_i,j}$ similarly to Equ. 16. Note that, the above process can be sped up using the homomorphic property of KZG commitment [47]. With valid public keys for each individual WSP, the VE can check the validity of individual credentials $Tok_{w_k}$ as the verification of a multi-message PS signature. If an invalid credential is detected, the VE can upload the corresponding evidence to the blockchain and claims back the deposit.

WSPs can detect if the VE's request and off-chain messages are inconsistent. Specifically, if $Sign_{sk_{v_i}}(m'_t)$ is valid but $H_0(\mathcal{S}_A)$ is incorrect, the WSP can send the corresponding evidence to the blockchain. The blockchain verifies the correctness of the signatures, and sends a notice about the VE misbehavior if it is confirmed. As a result, the request with $rid$ is withdrawn and the VE's deposit is transferred to the blockchain.

*Discussions:* (1) External accountability enforcement for misbehaving WSPs can be considered for different SAG services; (2) *SAG-BC* utilizes off-chain communications in the CCI to improve the on-chain efficiency. In case of message non-delivery (by VEs or WSPs), any receipt can require the

message to be sent on-chain using the encrypted communications under the receipt's public encryption key. Timely responses on-chain can be further achieved with time-locked deposit as incentives; (3) The subscription server manages service subscription fees and user identity (to communicate with the blockchain), but does not participate in the credential issuance. The subscription payments can also be checked by corresponding WSPs to make sure no free-riding happens. The role of the subscription server can be replaced by a distributed committee between WSPs with on-chain or off-chain payment channels, which requires independent research efforts.

## VI. Security Analysis

In this section, we analyze the security properties of *SAG-BC*. First, we analyze the security of the DSS and CCI, and the generated credential. Then, we discuss the verifiability of the DSS and CCI.

### A. DSS Security

DSS security ensures that public parameters and PS public keys for the credential issuance are correct and random.

For the public parameters, (1) the generations of elliptic curve parameters $\mathbb{G}$, base generators $g, \tilde{g}$, and a set of structured generators $\tilde{\mathcal{C}}$ are secure, which can be instantiated from a multi-party computation protocol [45]; (2) For public encryption keys of service providers, each individual WSP selects a secret key and computes the public key with a ZKP of correct constructions. Due to the security of ZKP, the well-formedness of the public key can be ensured.

For PS public keys, we use verifiable secret sharing to distribute shares of $(x, y_0, \{y_{i,j}\})$ to WSPs and generate the corresponding public keys [29], [47]. Each WSP $w_i$ commits to a polynomial using KZG commitment [44], [47], [48] and evaluates the polynomial at points from 0 to $n$. Due to the *binding* property of the KZG commitment, the WSP cannot generate two evaluations at the same position. As a result, any other WSP $w_j$ who receives the $(j, P_{w_i}(j))$ can verify the correctness of the polynomial evaluations. With this verifiability property of the shares, service providers do not quit the setup process in SAGVN under our security model. At the same time, the polynomial commitments are encrypted under a sematic secure scheme and the revelation is deterministic due to the ZKP of $\pi_{w_i,D}$. An optional re-randomization of public keys can also be considered [51]. Therefore, once all service providers finish the DSS, correct and random public keys are generated on the blockchain.

### B. CCI Security

With the secure setup of the system parameters and PS public keys, WSPs can collaboratively generate service credentials $Cerd_{v_i}$ for a VE $v_i$. First, $T_v = H_1(ID_{v_i}||pk_{v_i})^{id_{v_i}}$ is securely generated by the VE with a proof of knowledge $\pi_{v_i}$. $\pi_{v_i}$ is also verified by the blockchain to ensure that $T_v$ and $T'_v$ are correctly computed. In the PS signature, it is essential for signers to choose a random generator for each signature, which is achieved by using $H_1$ as a random oracle to output a generator $h_{v_i}$. At the same time, a random message $m'$ is also generated from $H_1$ to be included in generating credentials. Second, the blockchain records and checks the uniqueness of $h_{v_i}$, $T_v$, and $T'_v$. This ensures WSPs have a consistent view of VE credential requests and no messages are signed on the same generator before a partial token is computed.

After a VE's credential request is verified and stored on the blockchain, the VE can require the credential $Cerd_{v_i}$ from a random set of WSPs, denoted by $\mathcal{S}_W$. Note that, there are at least $t$ WSPs in $\mathcal{S}_W$. Then, each WSP $w_i$ in $\mathcal{S}_W$ can generate a partial token $Tok_{w_i}$ with its shares of $(x, y_0)$ and shares of secret keys $y_{i,j}$ of $AP_{i,j} \in S_A$. Due to the correctness of the secret sharing [26], [27], $t$ valid partial credentials can be aggregated to recover a subscription credential. Moreover, a computationally-bounded adversary without the knowledge of enough shares of secret keys cannot forge a valid credential.

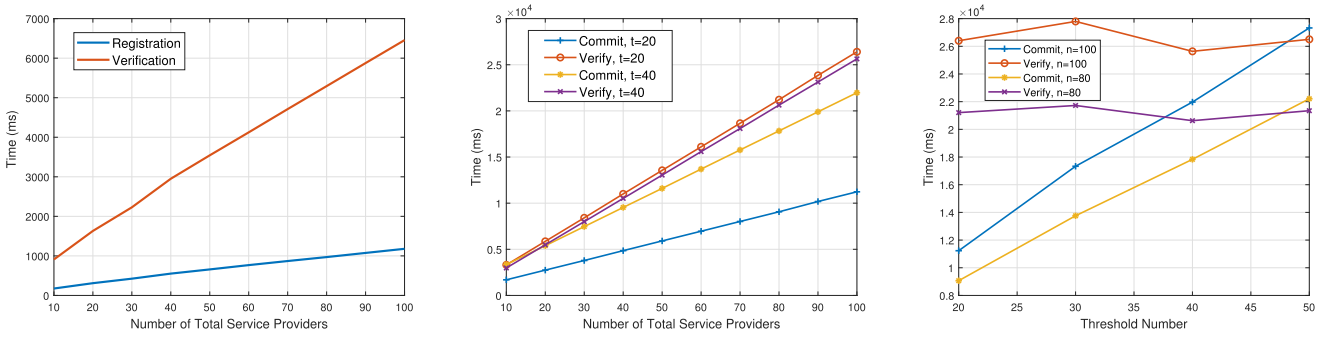
### C. Credential Security

The subscription credential is a PS signature on the VE-chosen secret $id_{v_i}$. Later, VEs with the knowledge of $id_{v_i}$ and the credential can prove knowledge of a signature via a ZKP protocol at corresponding APs. First, during the generation of the credential, VEs generate $T_v = h_{v_i}^{id_{v_i}}$ with a proof $\pi_{v_i}$, to be signed by WSPs. An efficient adversary cannot extract $id_{v_i}$ from $T_v$. Note that, the generator $h_{v_i}$ in the credential must be randomly selected. In *SAG-BC*, we require service providers to compute the $h_{v_i}$ from a hash function based on VEs' public information, which does not affect the security of the signature [31]. Second, in the process of proof knowledge of the signature, without the knowledge of $id_{v_i}$ of the credential, an efficient adversary cannot generate a valid ZKP, unless the adversary can either forge a PS signature or break the soundness property of the ZKP [40].

### D. Verifiability

We utilize the blockchain to promote honest collaboration between VEs and WSPs. Informally, the blockchain is a distributed ledger maintained by WSPs, that (1) enables consistent and immutable shared ledger storage among participants, and (2) guarantees secure and timely leger state update [52]. More specifically, the DSS and CCI contracts ensure that only valid messages (transactions) from VEs or WSPs can change the ledger state, where 'valid' means that messages come from an authenticated source and terms defined in the two contracts are satisfied. In a consortium blockchain, unless most of the blockchain nodes are malicious, the security of the blockchain is preserved [35].

In the DSS, first, each WSP publishes its commitment of a polynomial and encrypted polynomial evaluations to other WSPs on the contract. Due to the security of the DSS and the ZKP, no efficient adversary can forge a valid polynomial commitment or valid evaluations of the polynomial. Second, the commitments of polynomials and shares are encrypted using ElGamal encryption on the blockchain. The encryptions can later be verifiably decrypted by proving the same secret key between the decryption and the sender's public key. That is, the commitments and shares can be faithfully opened in




(a) DSS Registration (b) DSS Commit & Verify, $t = 20, 40$ (c) DSS Commit & Verify, $n = 80, 100$

Fig. 5. Off-chain computation overheads of DSS.

case of disputes. The decrypted polynomial commitments and evaluations can be verified on the blockchain. To this end, all operations of WSPs are undeniable and can be verified in the DSS. As a result, WSPs are motivated to behave honestly since detections of misbehavior can lead to the exclusion from providing SAG services and potential accountability enforcement.

In the CCI, all off-chain transmitted messages are signed by message senders, which can serve as non-repudiatable evidence of communications. For the verifiability of WSP behavior, first, an aggregated credential can be verified using the verification algorithm of PS signature. Second, if the aggregated credential is not correct, VEs can also verify individual credentials generated by WSPs. More specifically, WSPs reveal all $\tilde{C}'_{w_i,j}, \tilde{C}_{w_i,j}$ with the proof of knowledge $\pi_{w_i,j}$, which serve two purposes: (1) The blockchain can verify the correct evaluations at $(j, P_{w_i}(j))$; (2) Valid $\tilde{C}'_{w_i,j}$ can be aggregated to generate public PS keys for each WSP, that can be used to verify the partial credential. For the verifiability of VE behavior, VEs generate signatures on credential requests and calculate hashes of subscribed services, which cannot be forged due to the security of the signature and hash algorithms.

## VII. Performance Evaluation

In this section, we present off-chain and on-chain experimental results of *SAG-BC*.

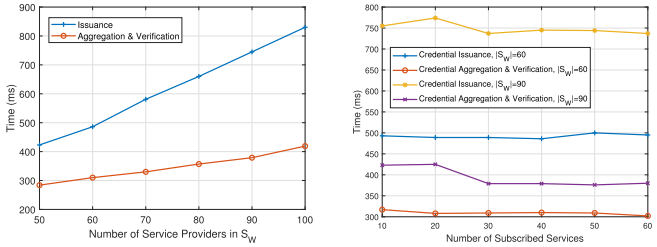
### A. Off-Chain Experiments

We test the off-chain operations in the DSS and CCI on a laptop with a 2.30 GHz processor and 8 GB memory. We implement the java pairing-based cryptography library (JPBC) [53] with the Type F curve. We test and report the computation overheads of DSS and CCI in Figure 5 and 6. We denote $n$ as the number of total service providers and $t$ as the threshold number. $S_A$ is the set of subscribed services and $S_W$ is the set of service providers for constructing a subscription credential.

In the registration, we measure the time when each service provider generates a public/private key pair with a proof and the verification of the proof. As shown in Figure 5a, the time cost for the registration/verification of the key pairs and proofs increases linearly with the number of service providers in the system. The results are feasible as it only takes a few seconds for 100 service providers. Moreover, although the cryptographic complexity for generation and verification of the key pairs is close, the verification in our experiments takes more time due to the use of 'isEqual' function in JPBC to determine if two elements are equal.

In the commit and verify phase of DSS, we measure the computation cost for the generation/verification of individual polynomial commitments from a single service provider. In Figure 5b and 5c, *commit* denotes the computation time when a service provider generates a polynomial commitment, evaluates the polynomial and generates the proofs for each other service provider. *Verify* denotes the computation time to verify $n$ polynomial evaluations and proofs. The performance is mainly affected by $n$ and $t$. It should be noted that the values of $t$ and $n$ in the experiments are set for performance evaluations. In Figure 5b, we fix $t = 20$ or $40$ and report the performance when $n$ increases. We can see that the time cost increases linearly with $n$. In Figure 5c, we fix $n = 80$ or $100$ and report the performance when $t$ increases. As we can see, the verification cost remains steady while commitment cost increases linearly with $t$. This is due to the use of the polynomial commitment that makes the verification succinct without being affected by $t$. By contrast, without the use of the polynomial commitment, each of $n$ participants needs to publish commitments of every coefficient in a degree-$(t-1)$ polynomial, which results in a $O(nt)$ computation (group exponentiations) and storage (group elements in $\mathbb{G}_2$) complexity for generating and verifying evaluations of a single polynomial. The time cost for encrypting a polynomial commitment in the Upload phase is around 12 milliseconds and the time costs for generating/verifying the decrypted commitment in the Reveal phase are around 15 or 60 milliseconds, respectively.

The CCI mainly involves credential requests by users, the credential issuance by service providers in $S_W$, and the credential aggregation and verification by users. For the credential request, we measure the generation and verification of $T_v$ and $T'_v$ in Equ. 18, which takes roughly 20 milliseconds. For the credential issuance, we measure the total time for all service providers in $S_W$ to calculate a partial credential (Equ. 21). As shown in Figure 6a, the number of subscribed services is set at 40 and the total time cost linearly increases with the number of service providers in $S_W$, which is still highly




(a) $|S_A| = 40$ (b) $|S_W| = 60, 90$

Fig. 6. Off-chain computation overheads of CCI.

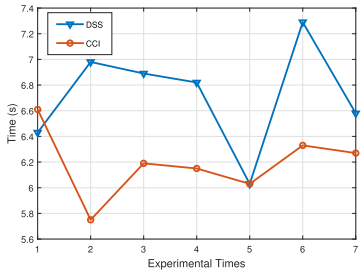



Fig. 7. On-chain response time of DSS/CCI.

efficient even for 100 providers (around 800 milliseconds in total). It should also be mentioned that the credential issuance can be computed parallelly by each provider. In Figure 6b, we fix $|S_W|$ at 60 or 90 and measure the impact of the number of subscribed services ($|S_A|$). The time costs remain almost the same with $|S_A|$ since secret keys of different services can be efficiently added in $\mathbb{Z}_p$. For the aggregation and verification, the time cost slightly increases with $|S_W|$ and is also not affected by $|S_A|$. This is because the additions of $\mathbb{Z}_p$ elements is much more efficient compared with the exponentiations in an elliptic curve. As a result, *SAG-BC* achieves the efficient multi-subscription credential without introducing additional overhead for credential generations and verifications.

### B. On-Chain Experiments

We set up a consortium blockchain network based on Hyperledger Fabric [35] on the same laptop. The network consists of a few peer nodes with a single ordering service. We use Java to implement the on-chain computations, which are packaged as chaincodes in Hyperledger Fabric. To support pairing-based cryptography in the chaincode, we adopt the Type F curve in JPBC, and include the JPBC library and curve parameters into the chaincode dependencies. Each peer node installs and approves the chaincode packages with the dependencies, and ensures that every node has the same version of chaincode package. Later, the peer nodes can call chaincode functions by sending transactions to the blockchain network.

We evaluate the performance of DSS and CCI. Since most computations are conducted off-chain, the dominant computations are to check Equ. 14 for the verification of polynomial evaluation at position 0 in DSS, and to check the correctness of Equ. 18 for the user registration in CCI. By using the same codes from our off-chain experiments, we set $n = 100$ in the experiments and generate the proofs off-chain. We embed public generators onto the chaincode and send the off-chain generated proofs via function calls. The peer nodes check the correctness of the proofs and send confirmations if the checks pass. After sufficient confirmations are received, the function call changes the ledger state. In Figure 7, we repeat the experiments 7 times and report the time difference between sending a function call and receiving a response. We can see that the response time is around 6 seconds for both verifications, which is highly dependent on the blockchain settings and consensus protocols. That is, the off-chain computations are negligible compared with the consensus protocols of the blockchain system in determining the overall response time. While the delay is feasible for credential management in SAGVN, our designs can be adapted to blockchain systems with more efficient consensus protocols.

In summary, the on/off-chain model for credential management in SAGVN only processes succinct information on the blockchain to reduce on-chain overheads. Moreover, *SAG-BC* is still efficient for credential generation and verification while embedding identity and multiple service subscription into a single credential.

## VIII. Conclusion

In this paper, we have proposed a blockchain-based credential management scheme in antenna array-enabled SAGVN, named *SAG-BC*. *SAG-BC* enables secure and distributed setup of system public parameters and collaborative credential issuance for anonymous authentications in SAGVN. On addressing the on-chain efficiency issue, *SAG-BC* has designed an on/off-chain communication protocol with succinct commitments and zero-knowledge proofs for verifiable operations of participants in SAGVN. With the exploration of a subscription-based service model, comprehensive architectural and protocol design for blockchain-based credential management, and thorough implementations and evaluations, *SAG-BC* can shed light on the research and practice of the service paradigm in SAGVN.

In the future, we will further explore the subscriber privacy preservation against service providers in the credential issuance phase, where service providers are not aware of the whole service subscriptions of VEs. To prevent leakage of subscription information from payments, anonymous payment channels can also be designed between users and service providers for service subscriptions in SAGVN.

## References

[1] H. Wu *et al.*, "Resource management in space-air-ground integrated vehicular networks: SDN control and AI algorithm design," *IEEE Wireless Commun.*, vol. 27, no. 6, pp. 52–60, Dec. 2020.

[2] S. Ghosh and D. Sen, "An inclusive survey on array antenna design for millimeter-wave communications," *IEEE Access*, vol. 7, pp. 83137–83161, 2019.

[3] Z. Xiao, L. Zhu, Z. Gao, D. O. Wu, and X. Xia, "User fairness non-orthogonal multiple access (NOMA) for millimeter-wave communications with analog beamforming," *IEEE Trans. Wireless Commun.*, vol. 18, no. 7, pp. 3411–3423, Jul. 2019.

[4] Z. Xiao, H. Dong, L. Bai, P. Xia, and X.-G. Xia, "Enhanced channel estimation and codebook design for millimeter-wave communication," *IEEE Trans. Veh. Technol.*, vol. 67, no. 10, pp. 9393–9405, Oct. 2018.

[5] H. Zhang, L. Song, Z. Han, and H. V. Poor, "Cooperation techniques for a cellular internet of unmanned aerial vehicles," *IEEE Wireless Commun.*, vol. 26, no. 5, pp. 167–173, Oct. 2019.

[6] M. F. Ozkoc, A. Koutsaftis, R. Kumar, P. Liu, and S. S. Panwar, "The impact of multi-connectivity and handover constraints on millimeter wave and terahertz cellular networks," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 6, pp. 1833–1853, Jun. 2021.

[7] Q. Yang, K. Xue, J. Xu, J. Wang, F. Li, and N. Yu, "AnFRA: Anonymous and fast roaming authentication for space information network," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 2, pp. 486–497, Jul. 2018.

[8] G. Neven, G. Baldini, J. Camenisch, and R. Neisse, "Privacy-preserving attribute-based credentials in cooperative intelligent transport systems," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, Nov. 2017, pp. 131–138.

[9] K. Xue, W. Meng, H. Zhou, D. S. L. Wei, and M. Guizani, "A lightweight and secure group key based handover authentication protocol for the software-defined space information network," *IEEE Trans. Wireless Commun.*, vol. 19, no. 6, pp. 3673–3684, Jun. 2020.

[10] B. Brecht and T. Hehn, "A security credential management system for V2X communications," in *Connected Vehicles*. Cham, Switzerland: Springer, 2019, pp. 83–115.

[11] D. Liu, H. Wu, J. Ni, and X. Shen, "Efficient and anonymous authentication with succinct multi-subscription credential in SAGVN," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 2863–2879, Feb. 2022.

[12] K. Gai, Y. Wu, L. Zhu, K.-K.-R. Choo, and B. Xiao, "Blockchain-enabled trustworthy group communications in UAV networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4118–4130, Jul. 2021.

[13] Z. Ma, J. Zhang, Y. Guo, Y. Liu, X. Liu, and W. He, "An efficient decentralized key management mechanism for VANET with blockchain," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 5836–5849, Jun. 2020.

[14] J. Zhang, J. Cui, H. Zhong, I. Bolodurina, and L. Liu, "Intelligent drone-assisted anonymous authentication and key agreement for 5G/B5G vehicular Ad-hoc networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 4, pp. 2982–2994, Oct. 2021, doi: 10.1109/TNSE.2020.3029784.

[15] A. Tomescu, V. Bhupatiraju, D. Papadopoulos, C. Papamanthou, N. Triandopoulos, and S. Devadas, "Transparency logs via append-only authenticated dictionaries," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2019, pp. 1299–1316.

[16] D. Liu, A. Alahmadi, J. Ni, X. Lin, and X. Shen, "Anonymous reputation system for IIoT-enabled retail marketing atop PoS blockchain," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3527–3537, Jun. 2019.

[17] S. Guo, X. Hu, S. Guo, X. Qiu, and F. Qi, "Blockchain meets edge computing: A distributed and trusted authentication system," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 1972–1983, Mar. 2020.

[18] A. Koutsos, "The 5G-AKA authentication protocol privacy," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroSP)*, Jun. 2019, pp. 464–479.

[19] S. Yao, J. Guan, Y. Wu, K. Xu, and M. Xu, "Toward secure and lightweight access authentication in SAGINs," *IEEE Wireless Commun.*, vol. 27, no. 6, pp. 75–81, Dec. 2020.

[20] M. Waqas, M. Ahmed, Y. Li, D. Jin, and S. Chen, "Social-aware secret key generation for secure device-to-device communication via trusted and non-trusted relays," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 3918–3930, Jun. 2018.

[21] W. Wang, Y. Chen, and Q. Zhang, "Privacy-preserving location authentication in Wi-Fi networks using fine-grained physical layer signatures," *IEEE Trans. Wireless Commun.*, vol. 15, no. 2, pp. 1218–1225, Feb. 2016.

[22] *Study on Security Aspects for LTE Support of V2X Services*, document TR 33.885, Version 2.0.0, 3GPP, 2017.

[23] *Security Architecture and Procedures for 5G System*, document TS 33.501, Version 0.8.0, 3GPP, 2018.

[24] P. Schindler, A. Judmayer, N. Stifter, and E. Weippl, "ETHDKG: Distributed key generation with ethereum smart contracts," Cryptol. ePrint Arch., Paper 2019/985, 2019. [Online]. Available: https://eprint.iacr.org/2019/985

[25] T. P. Pedersen, "A threshold cryptosystem without a trusted party," in *Proc. Workshop Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1991, pp. 522–526.

[26] P. Feldman, "A practical scheme for non-interactive verifiable secret sharing," in *Proc. 28th Annu. Symp. Found. Comput. Sci. (SFCS)*, Oct. 1987, pp. 427–438.

[27] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 1991, pp. 129–140.

[28] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure distributed key generation for discrete-log based cryptosystems," *J. Cryptol.*, vol. 20, no. 1, pp. 51–83, 2007.

[29] A. Tomescu *et al.*, "Towards scalable threshold cryptosystems," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2020, pp. 877–893.

[30] B. Schoenmakers, "A simple publicly verifiable secret sharing scheme and its application to electronic voting," in *Proc. CRYPTO*. Berlin, Germany: Springer, 1999, pp. 148–164.

[31] J. Camenisch, M. Drijvers, A. Lehmann, G. Neven, and P. Towa, "Short threshold dynamic group signatures," in *Proc. Int. Conf. Secur. Cryptogr. Netw.* Cham, Switzerland: Springer, 2020, pp. 401–423.

[32] A. Sonnino, M. Al-Bassam, S. Bano, S. Meiklejohn, and G. Danezis, "Coconut: Threshold issuance selective disclosure credentials with applications to distributed ledgers," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2019, pp. 1–15.

[33] W. Xu, W. Shi, F. Lyu, H. Zhou, N. Cheng, and X. Shen, "Throughput analysis of vehicular internet access via roadside WiFi hotspot," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3980–3991, Apr. 2019.

[34] J. Mendling *et al.*, "Blockchains for business process management-challenges and opportunities," *ACM Trans. Manage. Inf. Syst.*, vol. 9, no. 1, pp. 1–16, Feb. 2018.

[35] E. Androulaki *et al.*, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proc. 13th EuroSys Conf.*, Apr. 2018, pp. 1–15.

[36] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 2001, pp. 213–229.

[37] M. H. Au, W. Susilo, and Y. Mu, "Constant-size dynamic k-TAA," in *Proc. Int. Conf. Secur. Cryptogr. Netw.* Berlin, Germany: Springer, 2006, pp. 111–125.

[38] J. Camenisch and A. Lysyanskaya, "An efficient system for non-transferable anonymous credentials with optional anonymity revocation," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2001, pp. 93–118.

[39] D. Pointcheval and O. Sanders, "Short randomizable signatures," in *Proc. Cryptographers'Track RSA Conf.*, 2016, pp. 111–126.

[40] O. Sanders, "Efficient redactable signature and application to anonymous credentials," in *Proc. Int. Conf. Public-Key Cryptogr. (PKC)*, 2020, pp. 628–656.

[41] O. Sanders, "Improving revocation for group signature with redactable signature," in *Proc. Int. Conf. Public-Key Cryptogr. (PKC)*, 2021, pp. 301–330.

[42] P.-A. Fouque and D. Pointcheval, "Threshold cryptosystems secure against chosen-ciphertext attacks," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Berlin, Germany: Springer, 2001, pp. 351–368.

[43] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[44] A. Kate, "Distributed key generation and its applications," Ph.D. thesis, UWSpace, 2010. [Online]. Available: http://hdl.handle.net/10012/5285

[45] S. Bowe, A. Gabizon, and M. D. Green, "A multi-party protocol for constructing the public parameters of the pinocchio zk-SNARK," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Berlin, Germany: Springer, 2018, pp. 64–77.

[46] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inf. Theory*, vol. IT-31, no. 4, pp. 469–472, Jul. 1985.

[47] A. Kate, G. M. Zaverucha, and I. Goldberg, "Constant-size commitments to polynomials and their applications," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Berlin, Germany: Springer, 2010, pp. 177–194.

[48] A. Kate, Y. Huang, and I. Goldberg, "Distributed key generation in the wild," Cryptol. ePrint Arch., Paper 2012/377, 2012. [Online]. Available: https://eprint.iacr.org/2012/377

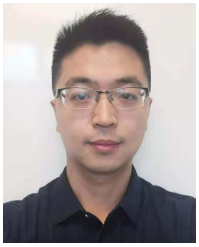
[49] A. Kate, A. Miller, and T. Yurek, "Brief note: Asynchronous verifiable secret sharing with optimal resilience and linear amortized overhead," 2019, *arXiv:1902.06095*.

[50] F. Brandt, "Efficient cryptographic protocol design based on distributed El Gamal encryption," in *Proc. Int. Conf. Inf. Secur. Cryptol.* Berlin, Germany: Springer, 2005, pp. 32–47.

[51] W. Neji, K. Blibech, and N. Ben Rajeb, "Distributed key generation protocol with a new complaint management strategy," *Secur. Commun. Netw.*, vol. 9, no. 17, pp. 4585–4595, Nov. 2016.

[52] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol," in *Proc. Annu. Int. Cryptol. Conf. (CRYPTO)*. Cham, Switzerland: Springer, 2017, pp. 357–388.

[53] A. De Caro and V. Iovino, "JPBC: Java pairing based cryptography," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun. 2011, pp. 850–855.



**Dongxiao Liu** (Member, IEEE) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Canada, in 2020. He is currently a Post-Doctoral Research Fellow with the Department of Electrical and Computer Engineering, University of Waterloo. His research interests include security and privacy in intelligent transportation systems, mobile networks, and blockchain.

**Huaqing Wu** (Member, IEEE) received the B.E. and M.E. degrees from the Beijing University of Posts and Telecommunications, Beijing, China, in 2014 and 2017, respectively, and the Ph.D. degree from the University of Waterloo, Waterloo, ON, Canada, in 2021. She worked as a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, McMaster University, from 2021 to 2022. She is currently an Assistant Professor with the Department of Electrical and Software Engineering, University of Calgary, Calgary, AB, Canada. Her current research interests include B5G/6G, space-air-ground integrated networks, the Internet of Vehicles, edge computing/caching, and artificial intelligence (AI) for future networking. She received the Best Paper Award from IEEE GLOBECOM 2018 and the Chinese Journal on Internet of Things 2020. She also received the prestigious Natural Sciences and Engineering Research Council of Canada (NSERC) Postdoctoral Fellowship Award in 2021.

**Cheng Huang** (Member, IEEE) received the B.Eng. and M.Eng. degrees in information security from Xidian University, China, in 2013 and 2016, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2020. He was a Project Officer with the INFINITUS Laboratory, School of Electrical and Electronic Engineering, Nanyang Technological University, in July 2016. His research interests are in the areas of applied cryptography, and cyber security and privacy in the mobile networks.

**Jianbing Ni** (Senior Member, IEEE) received the B.E. and M.S. degrees from the University of Electronic Science and Technology of China, Chengdu, China, in 2011 and 2014, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2018. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering and the Ingenuity Labs Research Institute, Queen's University, Kingston, ON. His current research interests include applied cryptography and network security, with a focus on cloud computing, smart grid, mobile crowdsensing, and the Internet of Things.

**Xuemin (Sherman) Shen** (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990.

He is currently a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research interests include network resource management, wireless network security, the Internet of Things, 5G and beyond, and vehicular ad hoc and sensor networks. He is a Registered Professional Engineer of ON, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, a Chinese Academy of Engineering Foreign Member, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society. He received the Canadian Award for Telecommunications Research from the Canadian Society of Information Theory (CSIT) in 2021; the R. A. Fessenden Award in 2019 from IEEE, Canada; the Award of Merit from the Federation of Chinese Canadian Professionals (ON) in 2019; the James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society; the Joseph LoCicero Award in 2015 and Education Award in 2017 from the IEEE Communications Society; and the Technical Recognition Award from Wireless Communications Technical Committee in 2019 and AHSN Technical Committee in 2013. He has also received the Excellent Graduate Supervision Award in 2006 from the University of Waterloo and the Premier's Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. He served as the Technical Program Committee Chair/Co-Chair for IEEE GLOBECOM 2016, IEEE Infocom 2014, IEEE VTC 2010 Fall, and IEEE GLOBECOM 2007, and the Chair for the IEEE Communications Society Technical Committee on Wireless Communications. He is the President of the IEEE Communications Society. He was the Vice President for Technical & Educational Activities, the Vice President for Publications, the Member-at-Large on the Board of Governors, the Chair of the Distinguished Lecturer Selection Committee, and a member of the IEEE Fellow Selection Committee of the ComSoc. He served as the Editor-in-Chief for the IEEE INTERNET OF THINGS JOURNAL, *IEEE Network*, and *IET Communications*.