SwiftAuth: Conditional Privacy-Preserving UAV Authentication with Multi-Source Credential Aggregation and Non-Interactive Authorization

Mingwei Zeng, Qingyang Zhang, Jie Cui *Senior Member, IEEE*, Hong Zhong, and Fengqun Wang

***Abstract*—To address the challenges of interactive credential issuance and conditional privacy-preserving authentication for dynamic terminals accessing multiple third-party services, we propose a traceable and aggregatable tagged witness signature (TA-TWS). Building upon this scheme, we design an efficient and privacy-preserving anonymous authentication framework, termed SwiftAuth. SwiftAuth decouples the issuer's authorization phase from the user's credential acquisition process in traditional anonymous authentication systems by introducing an asynchronous credential distribution mechanism. This allows issuers to generate authorization components even when users are offline, while ensuring that only the designated user can derive the complete access credential. Furthermore, during the authentication phase, SwiftAuth reduces communication and verification overhead in concurrent multi-service access by compressing multiple credentials into a constant-size representation, and supports verifier-side batch verification across users. In addition, it provides conditional privacy by integrating a regulatory trapdoor, ensuring default anonymity for legitimate users while enabling authorities to trace malicious actors upon protocol violations. Finally, we formally prove that the proposed TA-TWS achieves unforgeability and witness hiding. Extensive experiments were conducted on multiple resource-constrained devices, and Aliyun servers, using the MIRACL Core cryptographic library, demonstrate the practicality and efficiency of our approach.**

***Index Terms*—Anonymous Authentication; Privacy Protection; Tagged Witness Signature; Multi-service Authorization**

I. Introduction

In recent years, the Internet of Things (IoT) [1] and highly dynamic mobile terminals, such as unmanned aerial vehicle (UAV) swarms [2], [3] and autonomous vehicles, have increasingly relied on the Subscription-as-a-Service [4] model to obtain essential services. In such multi-source service access scenarios, terminals must frequently and concurrently interact with multiple independent service providers to acquire critical capabilities, including high-precision positioning, real-time environmental perception, and dynamic airspace access [5]. A fundamental and still insufficiently addressed security challenge in this context is how to achieve efficient credential verification while balancing anonymity and accountability [6], [7]. On the one hand, terminals must conceal their real identities during frequent interactions to prevent identity privacy leakage [8]. On the other hand, regulatory authorities must retain the capability to trace malicious entities in cases of credential abuse or illegal resale [9].

This tension is particularly pronounced in highly dynamic and continuously interactive IoT environments. Unlike traditional static IoT systems characterized by periodic data reporting [10], modern terminals frequently and concurrently request real-time services from different providers. For instance, in vehicular networks or UAV swarms [11], terminals continuously interact with edge services for positioning, sensing, and scheduling. Exposing real identities in such high-frequency interactions would inevitably enable linkage of user behaviors and trajectories [12]. Therefore, it is essential to ensure strong anonymity during normal operations while preserving the ability for reliable identity tracing when violations occur.

Anonymous Credentials. As a classical cryptographic primitive for privacy-preserving authentication, Anonymous Credentials typically rely on interactive blind signature between users and issuers to obtain credentials [13], allowing users to demonstrate credential ownership to verifiers via Zero-Knowledge Proofs without revealing their identities [14]. To accommodate complex application scenarios, recent research has derived numerous advanced properties based on AC, such as aggregatable anonymous credentials supporting cross-issuer credential compression [5], issuer-hidden credentials that further conceal the issuer's identity [15], as well as threshold and delegatable anonymous credentials [16], [17]. However, traditional Anonymous Credentials inherently provide complete anonymity, meaning that not even the credential issuer can trace a user's actual identity during a presentation process [18]. Consequently, in regulated scenarios, accountability and tracing mechanisms must be explicitly layered on top of the system before real-world deployment.

Group Signature. To reconcile anonymity with accountability, group signatures provide a natural alternative [19]. In such schemes, a user can generate a signature on behalf of the entire group while remaining anonymous to verifiers, and the group manager can open the signature to reveal the signer's identity when necessary. However, despite extensive optimizations [20], group signatures still incur significant computation and communication overhead [21]. This limitation becomes more pronounced in high-concurrency scenarios where terminals access multiple services simultaneously, as the associated costs scale linearly with the number of interactions for both users and verifiers [22]. Consequently, they struggle to satisfy the low-latency requirements of large-scale, highly dynamic IoT or UAV systems.

M. Zeng, Q. Zhang, J. Cui, H. Zhong and F. Wang are with the Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, School of Computer Science and Technology, Anhui University, Hefei 230039, China, and the Anhui Engineering Research Center for IoT Security Technologies, Anhui University, Hefei 230039, China (Corresponding authors: Hong Zhong; e-mail: zhongh@ahu.edu.cn).

Beyond the tension between privacy and efficiency, a key limitation of existing schemes is their reliance on synchronous connectivity [9], requiring both users and issuers to be online during credential issuance. For example, anonymous credential systems rely on interactive blind signature protocols [5], while group signatures involve explicit joining procedures. As a result, when an issuer needs to grant credentials to users on short notice, it must first establish interaction with them [21]. Such interaction-heavy designs are ill-suited for network-constrained or highly dynamic environments, significantly limiting scalability and flexibility.

Why consider Tagged Witness Signatures. To overcome the aforementioned limitations in authorization, accountability, and efficiency, this work is built upon the following key insight: the tagged witness signature (TWS) scheme proposed by Baldimtsi et al. [23] ensures that only entities possessing the witness corresponding to a specific tag can generate valid signatures. Inspired by this property, we optimize TWS and integrate it with public key derivation techniques [24] to construct a novel anonymous token mechanism for authentication. Specifically, an issuer can generate a service tag and a blinded PS-style partial credential from the user's public key and a regulator-provided signing base without requiring the user to be online. The user then removes the blinding and derives the tag-specific secret locally to obtain the complete witness for generating a presentation signature, which a verifier checks using the service tag rather than the user's public key. The shared signing base enables aggregation of the PS-form credential components from multiple issuers, while our batch-verification procedure processes presentations from multiple users efficiently. By decoupling credential presentation from user's public keys, this mechanism ensures strict anonymity. Additionally, embedding a trapdoor within the tag introduces conditional traceability without compromising default anonymity. Our contributions are summarized as follows:

1) To address the reliance on synchronous interactions in existing anonymous authentication mechanisms, we propose a novel traceable and aggregatable tagged witness signature primitive (TA-TWS). The proposed TA-TWS enables issuers to unilaterally generate authorization tags even when users are offline, while users can locally derive the corresponding witness using their private keys in order to complete credential generation. This design reduces traditional multi-round interactive protocols to a unidirectional broadcast process, significantly improving scalability and deployment flexibility.
2) To tackle the issues of credential size inflation and linear growth of verification overhead in concurrent multi-service access scenarios, TA-TWS construction inherently supports cross-issuer aggregation. This enables a constant-size compressed representation and efficient presentation of multi-source credentials. In addition, it supports batch verification of multiple user requests, reducing the computational complexity of verification.
3) To reconcile privacy protection with regulatory compliance, the verification of TA-TWS signature relies solely on ephemeral tags generated by the issuer and does not require the user's public key, thereby ensuring user anonymity. Furthermore, by embedding the regulatory key into tag, we enable the regulatory authority to trace the identities of misbehaving users, ultimately achieving a rigorous balance between privacy and accountability.
4) We formally prove that our scheme satisfies unforgeability and witness hiding. A prototype implementation based on the MIRACL Core library and real-world experiments conducted on a UAV-cloud platform demonstrate that our scheme achieves substantial performance improvements over existing solutions. The source code has been released publicly at https://github.com/rookie-papers/TA-TWS.git.

A. Paper Outline

The remainder of this paper is organized as follows. Section II reviews related work. Section III covers cryptographic preliminaries. Section IV describes the design of the proposed TA-TWS scheme. Section V provides the security analysis, and Section VI reports experimental results and comparisons.

II. Related Work

In this section, we provide a comprehensive review of the existing literature closely related to our research.

A. Anonymous Credentials

In multi-issuer scenarios, the aggregatability and presentation efficiency of credentials have garnered significant attention. Connolly et al. [28] constructed attribute-based credentials based on structure-preserving signatures on equivalence classes (SPS-EQ) [29] , enhancing privacy and malleability through joint randomization. To overcome the performance bottleneck of presenting multi-source credentials, Hébant et al. [30] utilized aggregate signatures with randomizable tags, achieving presentation overhead independent of the number of attributes and issuers. Mir et al. [5] combined SPS-EQ with their proposed AtoSa signature to design aggregatable credentials supporting issuer hiding, enabling verifiers to perform batch authentication on multi-authority credentials and substantially reducing the global verification overhead.

Threshold anonymous credentials aim to eliminate single points of trust through decentralization. Li et al. [26] proposed a dynamic threshold system that supports flexible adjustment of the issuer threshold without resetting the system, while reducing the overhead of batch presentation. Shi et al.'s [31] threshold scheme maintains selective disclosure capabilities while supporting threshold identity tracing and credential revocation required for decentralized governance. Doerner et al. [32] constructed a highly efficient secure multi-party signing protocol based on BBS+ signatures, achieving threshold credential issuance with composable security. Additionally, Mir et al. [33] combined threshold delegatable subset predicate encryption to extend the concept of delegatable anonymous credentials [25], realizing a credential system that simultaneously supports threshold issuance and privilege delegation.

Anonymous credentials have also made remarkable progress in lightweight design and practical deployment. Chairattana-Apirom et al. [27] formalized the server-aided anonymous

TABLE I
COMPARISON WITH OTHER PROTOCOLS

Scheme	Anonymous Authentication	Accountability	Aggregatability	Batch Authentication	Non-interactive Authorization
Griffy et al. [25]	✔	✔	✘	✘	✘
DTACB [26]	✔	✘	✔	✔	✘
NTAT [9]	✔	✘	✘	✘	✘
Hecate [15]	✔	✘	✔	✔	✘
IhMA [5]	✔	✘	✔	✔	✘
ChairApirom et al. [27]	✔	✘	✘	✘	✘
TA-TWS (Ours)	✔	✔	✔	✔	✔

credentials model, achieving a lightweight, multi-use construction on elliptic curves. Addressing credential abuse, Durak et al. [9] formalized the concept of non-transferability for anonymous tokens, balancing user privacy with the need to prevent unauthorized transfer. Rosenberg et al. [34] developed a universal zero-knowledge protocol that directly transforms existing traditional electronic identity documents into anonymous credentials without requiring issuing authorities to modify their infrastructure. Similarly, Baldimtsi et al. [35] allowed users to leverage existing OpenID Connect tokens and zero-knowledge proofs to achieve privacy-preserving rapid authentication without introducing new trusted parties.

B. Anonymous Authentication

Xu et al. [36] and Ai et al. [37] proposed lightweight anonymous authentication schemes for IoT devices. By integrating elliptic curve cryptography, pseudonym mechanisms, and hash functions, their designs enhance security and privacy during data offloading. Jiang et al. [38] proposed a conditional anonymous authentication scheme that integrates signatures of knowledge with blockchain-based Bloom filters, enabling efficient batch verification and public key revocation. Along the same line, Al Sibahee et al. [39] designed an efficient authentication protocol based on the k-valued modified bilinear inverse Diffie-Hellman problem, which effectively mitigates common threats such as privileged insider and replay attacks. Furthermore, Zhang et al. [40] combined one-way hash functions with bilinear assumptions to develop a robust vehicle authentication scheme that significantly reduces communication overhead while enhancing semantic security. Targeting Spatial Information Networks (SINs) with highly dynamic topologies, Yang et al. [41] proposed a periodic k-time keyed-verification anonymous credential scheme. Their design enables satellite nodes to support seamless satellite–ground handovers and anonymous communication, while strictly limiting the number of allowed authentications. In addition, within traditional communication networks, Du et al. [42] introduced an end-to-end anonymous authentication architecture, thereby effectively blocking malicious unknown calls without requiring modifications to the underlying infrastructure.

To elucidate the core distinctions between our work and state-of-the-art schemes, we provide a multi-dimensional comparison of several representative protocols in Table I.

III. Preliminaries

To facilitate a clear understanding of the proposed traceable and aggregatable tagged witness signature, this section introduces the foundational concepts employed in our construction.

A. Bilinear Pairings

Let $\mathbb{G}$ and $\tilde{\mathbb{G}}$ be multiplicative cyclic groups of prime order q, and let $\mathbb{G}_T$ be a target group of the same order. Let g and $\tilde{g}$ be generators of $\mathbb{G}$ and $\tilde{\mathbb{G}}$, respectively. A bilinear pairing is a map $e : \mathbb{G} \times \tilde{\mathbb{G}} \to \mathbb{G}_T$ satisfying the following properties:

1) **Bilinearity:** For all $u \in \mathbb{G}$, $v \in \tilde{\mathbb{G}}$, and $a, b \in \mathbb{Z}_q$, the bilinearity property ensures that the following equation $e(u^a, v^b) = e(u^{ab}, v) = e(u, v^{ab}) = e(u, v)^{ab}$ holds.
2) **Non-degeneracy:** The bilinear map e is non-trivial, if g and $\tilde{g}$ are generators of $\mathbb{G}$ and $\tilde{\mathbb{G}}$ respectively, then $e(g, \tilde{g})$ generates $\mathbb{G}_T$. Equivalently, for any $u \in \mathbb{G} \setminus \{1_{\mathbb{G}}\}$, the map $e(u, \cdot)$ is not the constant map to $1_{\mathbb{G}_T}$.
3) **Computability:** There exists an efficient algorithm to compute $e(u, v)$ for any $u \in \mathbb{G}$ and $v \in \tilde{\mathbb{G}}$.

B. Pointcheval-Sanders (PS) Signatures

The PS signature is an efficient, pairing-based short signature scheme [43]. Its inherent capability for seamless signature randomization makes it an ideal cryptographic building block for privacy-preserving applications such as anonymous credentials. A PS signature consists of the following algorithms:

- $\mathsf{Setup}(1^\lambda)$: On input a security parameter 1^λ, output the bilinear group parameters $bp = (q, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, e, g, \tilde{g})$, where $e : \mathbb{G} \times \tilde{\mathbb{G}} \to \mathbb{G}_T$ is a bilinear map over prime order q groups, with generators $g \in \mathbb{G}$ and $\tilde{g} \in \tilde{\mathbb{G}}$.
- $\mathsf{KeyGen}(1^\lambda)$: Choose random scalars $a, b \in \mathbb{Z}_q^*$ and compute $\tilde{A} = \tilde{g}^a$ and $\tilde{B} = \tilde{g}^b$. The public key is $pk = (\tilde{A}, \tilde{B})$, and the secret key is $sk = (a, b)$.
- $\mathsf{Sign}(sk, m)$: Given a message $m \in \mathbb{Z}_q^*$, select a random base $h \in_R \mathbb{G}^*$ and compute the signature $\sigma_1 = h, \sigma_2 = h^{a+b\cdot m}$. Finally, output the signature $\sigma = (\sigma_1, \sigma_2)$.
- $\mathsf{Verify}(pk, m, \sigma)$: Given $\sigma = (\sigma_1, \sigma_2)$, check if $\sigma_1 \neq 1_{\mathbb{G}}$, and then verify the equation $e(\sigma_1, \tilde{A} \cdot \tilde{B}^m) \stackrel{?}{=} e(\sigma_2, \tilde{g})$. Output 1 if the equation holds; otherwise, output 0.

Pointcheval and Sanders [43] have proved that the unforgeability of the PS signature scheme reduces to the Lysyanskaya-Rivest-Sahai-Wolf (LRSW) hardness assumption.

C. Tagged Witness Signatures (TWS)

A tagged witness signature [23] allows a user to sign a message by demonstrating the ability to obtain a secret witness w corresponding to a public tag t. For a predefined public predicate $P(t, w) = 1$, a TWS consists of three algorithms:

- $\mathsf{Gen}(1^\lambda, P) \to \mathsf{pk}$: Takes a security parameter λ and the predicate P, outputting the global public key pk.
- $\mathsf{Sign}(\mathsf{pk}, t, w, m) \to \sigma$: Given the public key pk, tag t, witness w, and message m, it outputs a signature σ strictly when the witness satisfies $P(t, w) = 1$.
- $\mathsf{Verify}(\mathsf{pk}, t, m, \sigma) \to \{0, 1\}$: Deterministically evaluates the tuple $(\mathsf{pk}, t, m, \sigma)$, outputting 1 (accept) if the signature is valid, and 0 (reject) otherwise.

IV. The proposed TA-TWS scheme

In this section, we present the detailed construction of our proposed traceable and aggregatable tagged witness signature (TA-TWS). We formalize the syntax of TA-TWS and define the threat model along with the desired security goals.

A. Syntax

Let $P : \mathcal{T} \times \mathcal{W} \to \{0, 1\}$ be a public predicate that takes a tag $T \in \mathcal{T}$ and a witness $w \in \mathcal{W}$ as inputs, outputting 1 if the witness is valid for the tag, and 0 otherwise. Our TA-TWS for the predicate P consists of a tuple of PPT algorithms:

$$\Pi = (\mathsf{Setup}, \mathsf{KGen}, \mathsf{TagGen}, \mathsf{WitGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Trace})$$

The formal definition of each algorithm is as follows:

$\mathsf{Setup}(1^\lambda) \to pp$**:** A probabilistic algorithm that takes a security parameter 1^λ as input. It outputs the system-wide public parameters pp. Implicitly, pp is provided as an input to all subsequent algorithms.

$\mathsf{KGen}(pp) \to (\mathcal{SK}, \mathcal{PK})$**:** A probabilistic algorithm that takes the public parameters pp as input. It outputs the universal sets of secret keys $\mathcal{SK}$ and public keys $\mathcal{PK}$ for all participating entities in the system.

$\mathsf{TagGen}(pp, \mathcal{PK}') \to T$**:** A probabilistic algorithm that takes the public parameters pp and a specific subset of public keys $\mathcal{PK}' \subseteq \mathcal{PK}$ as input. It outputs a dynamic tag T.

$\mathsf{WitGen}(pp, \mathcal{SK}', T, \mathcal{PK}') \to w$**:** A probabilistic algorithm that takes the system public parameters pp, a subset of secret keys $\mathcal{SK}' \subseteq \mathcal{SK}$, a specific tag T, and a subset of public keys $\mathcal{PK}' \subseteq \mathcal{PK}$ as input. It outputs a witness w corresponding to the tag T.

$\mathsf{Sign}(pp, w, T, msg) \to \Sigma$**:** A probabilistic algorithm that takes pp, a witness w, a tag T, and a message msg as input. It outputs a signature Σ under the tag T.

$\mathsf{Verify}(pp, T, msg, \Sigma) \to \{0, 1\}$**:** A deterministic algorithm that takes pp, a tag T, a message msg, and a signature Σ as input. It outputs 1 if Σ is a valid signature on the message msg under the tag T, and 0 otherwise.

$\mathsf{Trace}(pp, \mathcal{SK}', T, \mathcal{PK}') \to pk^* \cup \{\perp\}$**:** A deterministic algorithm that takes the public parameters pp, a specific subset of secret keys $\mathcal{SK}' \subseteq \mathcal{SK}$, a tag T, and a target subset of public keys $\mathcal{PK}' \subseteq \mathcal{PK}$ as input. It outputs a specific public key $pk^* \in \mathcal{PK}'$ representing the real identity of the signer if tracing is successful, or $\perp$ otherwise.

B. Threat Model and Trust Assumptions

We consider a PPT adversary that controls the public communication channels. The adversary can observe public interactions among users, issuers, and verifiers, and eavesdrop on, modify, inject, replay, delay, or discard service authorization data and TA-TWS signatures. It can adaptively select services, tag-generation requests, and request messages based on previous interactions. Its view includes public parameters, public keys, service tags, and valid TA-TWS signatures from different sessions. It may collect tags issued to the same user by different issuers and their corresponding signatures. Credential issuance need not be synchronous, and the adversary may interleave queries across users, issuers, and sessions.

The trusted RA allocates credential bases and performs authorized tracing according to the protocol. Issuers honestly generate tags and credentials for registered users. The long-term keys of the RA, issuers, and users are honestly generated, and the experiment does not reveal their private keys or the randomness used to generate tags. Through the witness oracle, the adversary may obtain complete witnesses for selected tags, including other tags belonging to the same user as the eventual target tag. Such leakage is limited to the corresponding tags and does not constitute disclosure of the user's long-term private key. The restrictions on issuance under the same base and the allocation rules for shared bases are specified in the construction and followed by the security experiment.

We examine the adversary's ability to generate a valid TA-TWS for a new combination of a tag list and a message when witnesses for some tags have been revealed, as well as the witness-hiding property of valid TA-TWS signatures for protected witnesses. The adversary's precise winning conditions are specified in the following security experiments.

C. Security Definitions

In this section, we formalize the security model of our proposed TA-TWS, comprising completeness, unforgeability, and witness hiding. The foundational formulations of these properties closely align with the TWS framework introduced by Baldimtsi et al. [23]. Completeness requires that any signature produced according to the protocol specification by honest parties is accepted by the verification algorithm.

Definition 1 (Completeness)**.** *Let Π be a TA-TWS scheme. We say that Π achieves completeness if, for any sufficiently large security parameter λ, any valid key subsets $\mathcal{SK}' \subseteq \mathcal{SK}, \mathcal{PK}' \subseteq \mathcal{PK}$, and any arbitrary message $msg \in \{0,1\}^*$, the following condition holds with overwhelming probability:*

$$\Pr\left[\mathsf{Verify}(\mathcal{I}) = 1 \;\middle|\; \begin{array}{l} pp \leftarrow \mathsf{Setup}(1^\lambda) \\ (\mathcal{SK}, \mathcal{PK}) \leftarrow \mathsf{KGen}(pp) \\ T \leftarrow \mathsf{TagGen}(pp, \mathcal{PK}') \\ w \leftarrow \mathsf{WitGen}(pp, \mathcal{SK}', T, \mathcal{PK}') \\ \Sigma \leftarrow \mathsf{Sign}(pp, w, T, msg) \\ \mathcal{I} := (pp, T, msg, \Sigma) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

To model unforgeability, we adopt the existential unforgeability under chosen tag and message attack (EUF-CTMA) paradigm. We define the adversary's advantage mathematically, followed by the specific interactive rules of the game.

Definition 2 (Unforgeability)**.** *Let* Π *be a TA-TWS scheme. The advantage of a probabilistic polynomial-time (PPT) adversary* $\mathcal{A}$ *is defined as follows, we say* Π *is EUF-CTMA secure if* $\mathbf{Adv}^{\textit{EUF-CTMA}}_{\Pi,\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$ *for any PPT adversary.*

$$\mathbf{Adv}^{\textit{EUF-CTMA}}_{\Pi,\mathcal{A}}(\lambda) := \Pr\left[\mathbf{Exp}^{\textit{EUF-CTMA}}_{\Pi,\mathcal{A}}(1^\lambda) = 1\right].$$

$\underline{\mathbf{Exp}^{\mathrm{EUF-CTMA}}_{\Pi,\mathcal{A}}(1^\lambda)}$

$pp \leftarrow \mathsf{Setup}(1^\lambda)$; $(\mathcal{SK}, \mathcal{PK}) \leftarrow \mathsf{KGen}(pp)$
$Q_H, Q_c, Q_w, Q_s, T_{\mathrm{list}} \leftarrow \emptyset$; clear the issuance cache
$(\mathcal{T}^*, msg^*, \Sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathrm{Tag}}, \mathcal{O}^{\mathrm{Cred}}, \mathcal{O}^{\mathrm{Wit}}, \mathcal{O}^{\mathrm{Sign}}, \mathcal{O}^{\mathrm{H}}}(pp, \mathcal{PK})$
If $\mathcal{T}^*$ is not a valid aggregate tag list, **return** 0
return 1 if and only if $\mathsf{Verify}(pp, \mathcal{T}^*, msg^*, \Sigma^*) = 1 \quad \wedge$
$(\mathcal{T}^*, msg^*) \notin Q_s \quad \wedge \quad \exists\, T_i^* \in \mathcal{T}^* : T_i^* \notin Q_w$;

$\underline{\mathcal{O}^{\mathrm{Tag}}(U, i, \mathrm{payload}, st)}$

If (U, i, st) is invalid, **return** $\perp$
$T \leftarrow \mathsf{TagGen}(pp, \mathcal{PK}'_{U,i})$
$T_{\mathrm{list}} \leftarrow T_{\mathrm{list}} \cup \{(T, U, i, st)\}$
return T
(U, i, st) identifies the registered user, the issuer, and the credential issuance round, respectively.

$\underline{\mathcal{O}^{\mathrm{Wit}}(T)}$

$w \leftarrow \mathsf{WitGen}(T)$
If $w = \perp$, **return** $\perp$
$Q_w \leftarrow Q_w \cup \{T\}$
return w

$\underline{\mathcal{O}^{\mathrm{H}}(j, \mathrm{input})}$

If (j, input) is recorded in Q_H, **return** $Q_H[j, \mathrm{input}]$
$y \xleftarrow{\$} \mathrm{Range}(H_j)$
$Q_H[j, \mathrm{input}] \leftarrow y$
return y
Here, input denotes the input to the hash function H_j, with j ranging over $\{1, 2, 3, 4, \mathrm{blind}\}$.

$\underline{\mathcal{O}^{\mathrm{Cred}}(T)}$

If $(T, U, i, st) \notin T_{\mathrm{list}}$, **return** $\perp$
Retrieve (U, i, st) from T_{list}, then compute $m_i \leftarrow H_2(T)$ and $H \leftarrow f(rsk, \widetilde{PK}_U, st)$
If $(i, H, T', \sigma_i) \in C_{\mathrm{list}}$ and $T' = T$, **return** (T, H, σ_i)
If $(i, H, T', \sigma_i) \in C_{\mathrm{list}}$ and $T' \neq T$, **return** $\perp$
$h \leftarrow H_{\mathrm{blind}}(\widetilde{PK}_U^{a_i})$
$\sigma_i \leftarrow H^{h(a_i + b_i m_i)}$
$C_{\mathrm{list}} \leftarrow C_{\mathrm{list}} \cup \{(i, H, T, \sigma_i)\}$
$Q_c \leftarrow Q_c \cup \{T\}$
return (T, H, σ_i)

$\underline{\mathcal{O}^{\mathrm{Sign}}(\mathcal{T}, msg)}$

If $\mathcal{T}$ is invalid, **return** $\perp$
For each $T_i \in \mathcal{T}$:
$w_i \leftarrow \mathsf{WitGen}(T_i)$
If $w_i = \perp$ for some i, **return** $\perp$
$\Sigma \leftarrow \mathsf{Sign}(pp, \{w_i\}, \mathcal{T}, msg)$
If $\Sigma = \perp$, **return** $\perp$
$Q_s \leftarrow Q_s \cup \{(\mathcal{T}, msg)\}$
return Σ

Fig. 1. The EUF-CTMA security experiment for TA-TWS.

In the EUF-CTMA experiment, the challenger runs Setup and KGen and provides the adversary with the public parameters and all participants' public keys. The adversary may adaptively query the tag, credential, and witness oracles to obtain service tags for specified registered users and the corresponding partial credentials or complete witnesses. It may also query the signing oracle to obtain TA-TWS signatures for chosen aggregate tag lists $\mathcal{T}$ and request messages msg.

Successful credential, witness, and signing queries add the corresponding tag to Q_c, the tag to Q_w, and $(\mathcal{T}, msg)$ to Q_s, respectively. Internal credential issuances by the witness and signing oracles are also recorded in Q_c, whereas internal witness use does not update Q_w. Queries returning $\perp$ do not update Q_w or Q_s, but prior issuances remain in Q_c. Repeated random oracle queries return consistent answers.

Thus, Q_w records tags for which complete witnesses have been returned to the adversary, Q_c records tags for which issuers have generated credentials, and Q_s records the tag-list and request-message pairs $(\mathcal{T}, msg)$ for which TA-TWS signatures have been returned to the adversary.

The experiment allows complete witness queries but does not reveal users' long-term secret keys or the randomness used to generate tags. The adversary ultimately outputs a valid aggregate tag list $\mathcal{T}^*$, a message msg^*, and a TA-TWS signature Σ^*. The experiment outputs 1 if and only if

$$\mathsf{Verify}(pp, \mathcal{T}^*, msg^*, \Sigma^*) = 1,$$
$$\text{subject to } (\mathcal{T}^*, msg^*) \notin Q_s \wedge \exists T_i^* \in \mathcal{T}^* : T_i^* \notin Q_w.$$

This condition permits witness queries for some aggregate components but requires that the complete witness of at least one component has not been returned to the adversary. The adversary's advantage is the probability that the experiment outputs 1. The scheme is EUF-CTMA secure if this probability is negligible for every PPT adversary.

Witness hiding ensures that a signature conceals the underlying witness employed during its creation. We formulate this property through an indistinguishability framework.

Definition 3 (Witness Hiding)**.** *Let* Π *be a TA-TWS scheme. The advantage of a PPT adversary* $\mathcal{A}$ *in distinguishing the real experiment from the simulated experiment is defined as follows, and we say that* Π *satisfies witness hiding if* $\mathbf{Adv}^{\textit{WH}}_{\Pi,\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$ *for any PPT adversary* $\mathcal{A}$*.*

$$\mathbf{Adv}^{\textit{WH}}_{\Pi,\mathcal{A}}(\lambda) := \left|\Pr[\mathbf{Exp\text{-}Real}(1^\lambda) = 1] - \Pr[\mathbf{Exp\text{-}Sim}(1^\lambda) = 1]\right|.$$

$\underline{\mathbf{Exp\text{-}Real}_{\Pi,\mathcal{A}}(1^\lambda)}$	$\underline{\mathbf{Exp\text{-}Sim}_{\Pi,\mathcal{A}}(1^\lambda)}$
$pp \leftarrow \mathsf{Setup}(1^\lambda)$	$pp \leftarrow \mathsf{Setup}(1^\lambda)$
$(\mathcal{SK}, \mathcal{PK}) \leftarrow \mathsf{KGen}(pp)$	$(\mathcal{SK}, \mathcal{PK}) \leftarrow \mathsf{KGen}(pp)$
$b \leftarrow \mathcal{A}^{\mathcal{O}^{\mathrm{H}}(\cdot), \mathcal{O}^{\mathrm{Sign}}(\cdot,\cdot)}(pp, \mathcal{PK})$	$b \leftarrow \mathcal{A}^{\mathcal{S}^{\mathrm{H}}(\cdot), \mathcal{O}^{\mathrm{SimSign}}(\cdot,\cdot)}(pp, \mathcal{PK})$
return b	**return** b
$\underline{\mathcal{O}^{\mathrm{Sign}}(T, msg)}$	$\underline{\mathcal{O}^{\mathrm{SimSign}}(T, msg)}$
$w \leftarrow \mathsf{WitGen}(pp, \mathcal{SK}', T, \mathcal{PK}')$	// $\mathcal{S}$ *controls and programs* $\mathcal{S}^H$
$\Sigma \leftarrow \mathsf{Sign}(pp, w, T, msg)$	$\Sigma \leftarrow \mathcal{S}(pp, T, msg)$
return Σ	**return** Σ

Fig. 2. The Witness Hiding Experiments for TA-TWS.

The two indistinguishability experiments evaluating this property are orchestrated as follows. In $\mathbf{Expt\text{-}Real}$, the challenger executes the authentic protocol operations, responding to $\mathcal{A}$'s signature queries by extracting the real witness and running the legitimate signing algorithm. Conversely, in $\mathbf{Expt\text{-}Sim}$, the challenger acts as a simulator equipped with the privilege to program random oracles. It answers all signature queries by fabricating computationally indistinguishable signature views via trapdoor simulation, entirely bypassing the knowledge of the genuine witness. At the end of the interactive phase, $\mathcal{A}$ outputs a bit $b \in \{0, 1\}$ to predict which computational environment it is currently subjected to.

D. Our Construction

Table II summarizes the principal symbols used below. The subscript i identifies an issuer and its service tag, while j indexes user requests in batch verification.

TABLE II
PRINCIPAL NOTATION IN THE TA-TWS CONSTRUCTION.

Symbol	Meaning
$q, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_t, e$	Group order, source/target groups, bilinear map.
$X, \tilde{Y}, \hat{Z}$	Group generators; $\hat{Z} = e(X, \tilde{Y})$.
rsk, PK_R	Regulator's tracing secret and public key.
$a_i, b_i, \tilde{PK}_{S,i}$	Issuer i's signing scalars and public key.
$usk, \tilde{PK}_U$	User's long-term secret and public key.
f, st	Pseudorandom derivation function, issuance state.
$T_i, \hat{T}_{vk,i}$	Service tag and its public verification value.
$m_i = H_2(T_i)$	Credential message derived from service tag i.
$\sigma_i, T_{sk,i}, w_i$	Issuer credential, temporary signing key, and complete witness.
$\sigma_{agg}, \tilde{K}_{agg}$	Aggregated credentials and issuer public keys.
$\hat{E}, \hat{F}, \hat{R}$	Pairing values and commitment in the proof.
msg, Σ	Request message and its TA-TWS signature.
H_1, H_2, H_3, H_4	Hash functions for different purposes.

In this section, we present the detailed construction of the proposed TA-TWS and describe each algorithm in Fig. 3.

Setup. The regulator generates bilinear pairing parameters $bp \leftarrow (q, X, \tilde{Y}, \hat{Z}, e, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_t)$, where X and $\tilde{Y}$ generate $\mathbb{G}$ and $\tilde{\mathbb{G}}$, respectively, and $\hat{Z} = e(X, \tilde{Y})$ generates $\mathbb{G}_t$. The function $f(rsk, \tilde{PK}_U, st)$ is a keyed pseudorandom derivation that maps the regulator's key, a user's public key, and the issuance state to a point $H \in \mathbb{G}$. The $pp = (q, X, \tilde{Y}, \hat{Z}, e, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_t, f)$ is published as the public parameters.

Key Generation. Given the public parameters pp, the key generation and registration process is carried out by the Regulator, Issuers, and Users in a distributed manner:

- **Regulator Key Generation:** The regulator $\mathcal{R}$ generates a master view secret key $rsk \in_R \mathbb{Z}_q^*$ and publishes the corresponding public view key $PK_R = X^{rsk}$.
- **Issuer Key Generation:** Each issuer $\mathcal{I}$ randomly selects $(a, b) \in_R (\mathbb{Z}_q^*)^2$ as its secret key ssk and computes $\tilde{PK}_S = (\tilde{A} = \tilde{Y}^a, \tilde{B} = \tilde{Y}^b)$ as its the verification key. Then it registers $\tilde{PK}_S$ with the regulator.
- **User Key Generation:** Each user $\mathcal{U}$ generates a private key $usk \in_R \mathbb{Z}_q^*$ and computes $\tilde{PK}_U = \tilde{Y}^{usk}$ as its long-term public key. The user submits $\tilde{PK}_U$ to the regulator, who records it in a user registry $\mathbb{U}$ to enable traceability.

Tag Generation. When user with public key $\tilde{PK}_U$ requests a service credential, issuer selects $t \in_R \mathbb{Z}_q$ and computes $X_t = X^t$, $\tilde{Y}_t = \tilde{Y}^t$ and the tag verification public key $\hat{T}_{vk} = e(X^{H_1(X_t, \tilde{Y}_t, e(\tilde{PK}_R^t, \tilde{PK}_U))}, \tilde{PK}_U)$. Finally, the service tag is defined as $T_i = \{\tilde{PK}_S, \text{payload}, X_t, \tilde{Y}_t, \hat{T}_{vk}\}$, where payload denotes application-specific auxiliary data. For instance, it can be utilized to store information such as the credential's expiration time.

Witness Generation. The witness generation process involves cooperative interaction between the regulator, the issuer, and the user, following these sequential phases:

- *Regulator:* To initiate credential provisioning, the issuer forwards $\tilde{PK}_U$ to the regulator. Then, the regulator utilizes a pseudo-random function PRF (or a verifiable random function VRF [44]) to generate a auxiliary parameter $H = f(rsk, \tilde{PK}_U, st)$ and maintains it in a local database H_{list}, which ensures that different issuers utilize a unique and consistent H for the same user.
- *Issuer:* Upon receiving H, the issuer $\mathcal{I}_i$ computes a blinding factor $h = H(\tilde{PK}_U^{a_i})$ and generates a blinded PS-style partial credential $\sigma_i = H^{h \cdot (a_i + b_i \cdot m_i)}$, where $m_i = H_2(T_i)$.

Traceable and Aggregatable Tagged Witness Signature

Setup(1^λ) :

1: $\mathbb{G} =< X >$, $\tilde{\mathbb{G}} =< \tilde{Y} >$, $\mathbb{G}_t =< \hat{Z} = e(X, \tilde{Y}) >$.
2: Return $pp = (q, X, \tilde{Y}, \hat{Z}, e, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_t, f)$, where f is the pseudorandom derivation function, group parameters generated by $\mathsf{BPGen}(1^\lambda)$.

KGen(pp) :

1: Issuer: $ssk = (a, b)$, $\tilde{PK}_S = (\tilde{A} = \tilde{Y}^a, \tilde{B} = \tilde{Y}^b)$.
2: Opener select $rsk \in_R \mathbb{Z}_q^*$ and compute $PK_R = X^{rsk}$.
3: User select $usk \in_R \mathbb{Z}_q^*$ and compute $\tilde{PK}_U = \tilde{Y}^{usk}$.

TagGen$(pp, \tilde{PK}_U, \tilde{PK}_R)$:

1: Issuer select $t \in_R \mathbb{Z}_q$ and compute $X_t = X^t, \tilde{Y}_t = \tilde{Y}^t$.
2: Compute $\hat{T}_{vk} = e(X^{H_1(X_t, \tilde{Y}_t, e(\tilde{PK}_R^t, \tilde{PK}_U))}, \tilde{PK}_U)$.
3: **return** Tag $T = \{T_{exp}, X_t, \tilde{Y}_t, \hat{T}_{vk}, \tilde{PK}_S\}$.

WitGen$(pp, ssk, T, \tilde{PK}_U)$:

1: User forwards $\tilde{PK}_U$ to the regulator via the issuer.
2: Regulator computes $H = f(rsk, \tilde{PK}_U, st)$, records it in the list H_{list}, and returns it to the issuer.
3: Issuer computes certificate $\sigma_i = H^{a_i + b_i \cdot H_2(T_i)}$ to user.
4: User compute $v = H_1(X_t, \tilde{Y}_t, e(\tilde{PK}_R, \tilde{Y}_t^{usk}))$, $T_{sk} = X^{v \cdot usk}$, and verify the correctness of $\hat{T}_{vk}$ by verifing $e(T_{sk}, \tilde{Y}) \stackrel{?}{=} \hat{T}_{vk}$.
5: **return** Witness $w = (\sigma_i, T_{sk})$ for tag T.

Sign(w, pp, T, msg) :

1: Compute $\sigma_{agg} = \prod_{i=1}^n \sigma_i$ and $\tilde{K}_{agg} = \prod_{i=1}^n (\tilde{A}_i \cdot \tilde{B}_i^{H_2(T_i)})$.
2: Select r, t and compute $H' = H^r, \sigma' = (\sigma_{agg} \cdot H^t)^r$.
3: Compute $\hat{E} = e(\sigma', \tilde{Y}) \cdot e(H', \tilde{K}_{agg})^{-1}$ and $\hat{F} = e(H', \tilde{Y})$.
4: Select $k \in_R \mathbb{Z}_q^*$ and compute commitment $\hat{R} = \hat{F}^k$.
5: Compute $c = H_3(msg, \hat{R}, \tilde{K}_{agg}, H', \sigma')$ and $s = k + c \cdot t$.
6: Select $x \in_R \mathbb{Z}_q^*$ and compute $\hat{Z}_x = \hat{Z}^x$.
7: Compute $h = H_4(msg, \hat{Z}_x, \{\hat{T}_{vk,i}\})$, $\sigma_x = (\prod_{i=1}^n T_{sk,i})^h \cdot X^x$.
8: **return** signature $\Sigma = (H', \sigma', \hat{R}, s, \hat{Z}_x, \sigma_x)$.

Verify(pp, T, msg, Σ) :

1: Compute $\tilde{K}_{agg} = \prod_{i=1}^n (\tilde{A}_i \cdot \tilde{B}_i^{m_i})$, where $m_i = H_2(T_i)$.
2: Compute $\hat{F} = e(H', \tilde{Y})$ and $\hat{E} = e(\sigma', \tilde{Y}) \cdot e(H', \tilde{K}_{agg})^{-1}$.
3: Verify $\hat{F}^s \stackrel{?}{=} \hat{R} \cdot \hat{E}^c$;, where $c = H_3(msg, \hat{R}, \tilde{K}_{agg}, H', \sigma')$.
4: **return** 1 if the equation $e(\sigma_x, \tilde{Y}) \stackrel{?}{=} \hat{Z}_x \cdot (\prod_{i=1}^n \hat{T}_{vk,i})^h$ hold, where $h = H_4(msg, \hat{Z}_x, \{\hat{T}_{vk,i}\})$, else **return** 0.

Trace$(pp, rsk, T, \tilde{PK}_U^*)$:

1: **if** $\hat{T}_{vk} = e(X^{H_1(X_t, \tilde{Y}_t, e(X_t, \tilde{PK}_{U,i}^{*})^{rsk})}, \tilde{PK}_{U,i}^*)$ for $i \in \mathbb{U}$.
2: **return** $\tilde{PK}_{U,i}^*$, if above equation hold for $\tilde{PK}_{U,i}^*$.

Fig. 3. The Proposed Traceable and Aggregatable Tagged Witness Signature.

The issuer then transmits the tuple $\{T_i, H, \sigma_i\}$ to the user.

- *User:* The user computes $v = H_1(X_t, \tilde{Y}_t, e(\tilde{PK}_R, \tilde{Y}_t^{usk}))$ and derives the temporary signing key $T_{sk} = X^{v \cdot usk}$. It then verifies the tag value by checking $e(T_{sk}, \tilde{Y}) \stackrel{?}{=} \hat{T}_{vk}$. Finally, it computes the matching blinding factor $h = H(\tilde{A}_i^{usk})$ and de-blinds the component via $\sigma_i \leftarrow \sigma_i^{1/h}$.

To ensure security, the regulator never sends the same H to an issuer more than once. This can be enforced by maintaining the state information st. For example, if st records the validity period of H, the regulator only needs to track which issuers have already received it during that period. After deblinding, $(H, \sigma_i) = (H, H^{a_i + b_i m_i})$ has the PS signature form under issuer i's public key for $m_i = H_2(T_i)$. The tag-specific secret T_{sk} is an additional component of TA-TWS rather than part of the PS signature. Finally, the user assembles the witness $w = (\sigma_i, T_{sk})$ for the dynamic tag T_i.

Signature Generation. To generate an signature for a message msg, the user first aggregates the individual witnesses received from different issuers into a single group element $\sigma_{agg} = \prod_{i=1}^{n} \sigma_i$. Next, the user selects random scalars $r, t \in_R \mathbb{Z}_q^*$ and computes the randomized elements $H' = H^r$ and $\sigma' = (\sigma_{agg} \cdot H^t)^r$. Concurrently, the user reconstructs the aggregated issuer public key $\tilde{K}_{agg} = \prod_{i=1}^{n} (\tilde{A}_i \cdot \tilde{B}_i^{m_i})$, where $m_i = H_2(T_i)$. The user calculates $\hat{E} = e(\sigma', \tilde{Y}) \cdot e(H', \tilde{K}_{agg})^{-1}$ and $\hat{F} = e(H', \tilde{Y})$. To prove possession of a valid randomized witness without revealing it, the user selects $k \in_R \mathbb{Z}_q^*$ and computes the commitment $\hat{R} = \hat{F}^k$, and derives the challenge $c = H_3(msg, \hat{R}, \tilde{K}_{agg}, H', \sigma')$. The corresponding response is then computed as $s = k + c \cdot t$. Subsequently, the user selects $x \in_R \mathbb{Z}_q^*$, computes the randomized generator $\hat{Z}_x = \hat{Z}^x$, and derives the message hash $h = H_4(msg, \hat{Z}_x, \{\hat{T}_{vk,i}\})$. Finally, the user generates $\sigma_x = (\prod_{i=1}^{n} T_{sk,i})^h \cdot X^x$ and outputs the complete anonymous signature $\Sigma = (H', \sigma', \hat{R}, s, \hat{Z}_x, \sigma_x)$.

Verification. To verify a signature Σ on message msg with respect to the tags T, the verifier performs:

1) If $H' = \sigma' = \hat{R} = 1$, the signature verification fails.
2) First, the verifier reconstructs the aggregated public key $\tilde{K}_{agg} = \prod_{i=1}^{n} (\tilde{A}_i \cdot \tilde{B}_i^{m_i})$ using the provided tags, ensuring the aggregation logic matches the signing phase.
3) Next, it computes the values $\hat{E} = e(\sigma', \tilde{Y}) \cdot e(H', \tilde{K}_{agg})^{-1}$ and $\hat{F} = e(H', \tilde{Y})$. It then checks the validity of the proof by verifying if the equation $\hat{F}^s \stackrel{?}{=} \hat{R} \cdot \hat{E}^c$ holds, where the challenge is re-derived as $c = H_3(msg, \hat{R}, \tilde{K}_{agg}, H', \sigma')$.
4) Finally, the verifier computes $h = H_4(msg, \hat{Z}_x, \{\hat{T}_{vk,i}\})$ and checks the validity of the message signature by verifying the following equation. If all checks pass, the verification returns 1; otherwise, it returns 0.

$$e(\sigma_x, \tilde{Y}) \stackrel{?}{=} \hat{Z}_x \cdot \left(\prod_{i=1}^{n} \hat{T}_{vk,i} \right)^h$$

Note that once the legitimacy of a tag T has been successfully verified, the verifier can cache $\hat{T}_{vk}$ locally. For subsequent access requests associated with the same tag, the verifier can directly validate the components $(\hat{Z}_x, \sigma_x)$. Since the signature component σ_x supports batch verification, this caching mechanism significantly enhances the authentication efficiency, especially in high-concurrency scenarios.

Batch Verification. To handle massive simultaneous authorization requests from a users, the verifier can authenticate N signatures simultaneously. To prevent rogue-key attacks while maximizing computational efficiency, the verifier employs a small exponent test by sampling a random vector of 128-bit integers $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_N) \stackrel{R}{\leftarrow} \{0,1\}^{128}$.

First, the Verifier batches the zero-knowledge proofs to validate the unforgeability and privacy constraints:

$$e\left(\prod_{j=1}^{N} \left(H_j'^{s_j} \cdot \sigma_j'^{-c_j} \right)^{\delta_j}, \tilde{Y} \right) \cdot \prod_{j=1}^{N} e\left(H_j'^{\delta_j c_j}, \tilde{K}_{agg,j} \right) \stackrel{?}{=} \prod_{j=1}^{N} \hat{R}_j^{\delta_j}$$

Second, the verifier batches the core message signatures to authenticate the aggregated credentials:

$$e\left(\prod_{j=1}^{N} \sigma_{x,j}^{\delta_j}, \tilde{Y} \right) \stackrel{?}{=} \left(\prod_{j=1}^{N} \hat{Z}_{x,j}^{\delta_j} \right) \cdot \prod_{j=1}^{N} (\hat{T}_{vk_agg,j})^{\delta_j h_j}$$

where $\hat{T}_{vk_agg,j} = \prod_i \hat{T}_{vk,i}$ denotes the aggregated tag verification key in the target group $\mathbb{G}_t$ for the j-th user.

The aforementioned batch verification reduces the total number of expensive pairing operations from $5N$ under naive sequential verification to just $N + 2$. Furthermore, bounding the random multipliers δ_j to 128 bits effectively halves the underlying modular exponentiation overhead without compromising the 128-bit security level of the BLS12-381 curve.

Identity Tracing. If malicious behavior occurs, the regulator can reveal the real identity of the user. The regulator identifies the real user by iterating through the database of registered users $i \in \mathbb{U}$ and using the secret key rsk to verify $\hat{T}_{vk} \stackrel{?}{=} e(X^{H_1(X_t, \tilde{Y}_t, e(X_t, \tilde{PK}_{U,i}^*)^{rsk})}, \tilde{PK}_{U,i}^*)$. If a match is found, the user's real identity $\tilde{PK}_{U,i}^*$ is revealed. This mechanism ensures accountability without requiring the regulator to store all mappings between tags and user identities.

E. Privacy-preserving Anonymous Authentication Framework

To concretely demonstrate the application potential of the proposed TA-TWS in practical complex systems, this section constructs a privacy-preserving anonymous authentication framework using a UAV network as a case study. It is imperative to emphasize that our scheme possesses high versatility; it is not limited to UAV scenarios but can be seamlessly extended to any distributed architecture requiring strict access control and multi-source credential management, such as Vehicle-to-Everything networks, smart grids, and the Internet of Things (IoT). The overall system architecture and the interaction logic among entities are illustrated in Fig. 4.

Service subscription is a classic paradigm for enabling on-demand resource access in distributed networks. As highly dynamic mobile nodes, commercial UAVs frequently require concurrent access to multiple third-party services, such as high-precision navigation maps, real-time meteorological data, during complex mission executions. These services are typically highly time-sensitive and are distributedly managed by different service providers. To enforce strict access control

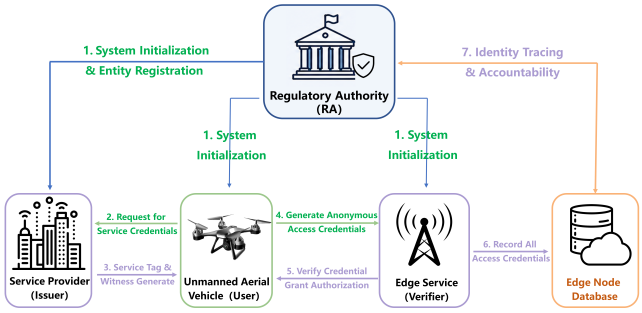


Fig. 4. The architecture and operational workflow of the proposed framework.

while effectively preserving UAV trajectory privacy and preventing credential theft, we design the following ecosystem which comprises four core physical entities, corresponding to the abstract roles in our algorithmic model: the regulatory authority, the service providers, the UAVs, and the base station. The specific operational workflow is detailed as follows:

System Initialization. The regulatory authority, typically assumed by a trusted national government department, acts as the paramount root of trust of the system. It executes the $\mathsf{Setup}(1^\lambda)$ algorithm to generate and publishes the global public parameters $pp = (q, X, \tilde{Y}, \hat{Z}, e, \mathbb{G}, \mathbb{G}_T, \mathbb{G}_t, f)$, establishing the cryptographic foundation for the entire network.

Entity Registration. Given the public parameters pp, the entities in the network complete registration in a distributed manner. The RA generates the master tracing secret key rsk and the public view key PK_R. Simultaneously, SPs and UAVs independently generate their respective key pairs, and apply for identity certificates from the RA, thereby cryptographically binding their public keys to their real-world physical identities.

Service Subscription. When a UAV initiates a service subscription, the SP first forwards its public key $\tilde{PK}_U$ to the RA. To prevent the UAV's trajectory from being maliciously linked due to presenting the same tag at different base stations, the system introduces a multi-tag rotation mechanism. Specifically, for any specific time epoch t (e.g., a specific month) requested by the UAV, the RA generate a set of mutually independent pseudonym values $H_{u,t}^{(k)} = f(rsk, \tilde{PK}_U||t||k)$, and stores them in a local database H_{list}.

Subsequently, based on this set of pseudonym values, the service provider invokes the $\mathsf{TagGen}(pp, \tilde{PK}_U, PK_R)$ to generate multiple dynamic service tags $T_{i,t}^{(k)}$ with identical validity periods for that specific epoch. These tags incorporate an application-specific payload designed to carry critical information, such as the service expiration time and the SP's public key. Upon receiving the tag set, the UAV utilizes usk to verify their legitimacy. When flying across different regions, the UAV can achieve trajectory unlinkability and anonymity by switching between different tags at various base stations.

Witness Extraction. Following tag generation, the service provider invokes the $\mathsf{WitGen}(pp, ssk, T, \tilde{PK}_U)$ protocol for each tag to compute a partial witness σ_i and distributes it to the UAV. Subsequently, the UAV utilizes its private key usk to locally derive the temporary signing key T_{sk} corresponding to the tag, thereby autonomously reconstructing the complete authorization witness $w_i = (\sigma_i, T_{sk})$.

Anonymous Access Authentication. During mission execution, a UAV often needs to concurrently request multiple independent services. Benefiting from the aggregation property of our scheme, the UAV does not need to present individual proofs sequentially. Instead, by executing the $\mathsf{Sign}(w, pp, \{T_i\}, msg)$ algorithm, it compresses multiple heterogeneous service credentials into a single, constant-size signature Σ, binding it to a specific access request message msg. This significantly reduces communication and authentication latency. Furthermore, if a UAV continuously operates within the coverage area of the same base station, it only needs to prove possession of the witness T_{sk}, eliminating the need to repeatedly verify the structural legitimacy of the tags.

Identity Verification. Base stations deployed at the network edge act as proxy verifiers, rapidly verifying the legitimacy of the UAV by invoking the $\mathsf{Verify}(pp, \{T_i\}, msg, \Sigma)$ algorithm. Our scheme not only supports the aggregated verification of multi-source credentials for a single node but also enables the batch verification of signatures from N different UAVs via the $\mathsf{BatchVerify}$. Notably, once a tag T_i is successfully verified for the first time, the base station can cache it. Subsequent requests associated with this tag can directly bypass redundant checks, further elevating system throughput.

Identity Tracing. Since the service tag only reveals a randomized ephemeral public key, the base station cannot infer the UAV's real identity from it. Meanwhile, the system enforces strong accountability for malicious behavior without compromising privacy in routine operations. If a entity detects a violation, it simply submits the transaction transcript to the registration authority. As the sole entity holding global user registration information, the RA invokes the $\mathsf{Trace}(pp, rsk, T_i, \tilde{PK}_U^*)$ algorithm, iterates over the PK_U database, and matches using the master tracing key rsk, thereby seamlessly revealing the real identity $\tilde{PK}_U^*$. This mechanism achieves ideal conditional privacy: it neither interferes with high-frequency authentication nor imposes significant storage overhead on system nodes. Moreover, the RA does not need to store large amounts of temporary tag information; it only needs to maintain users' public keys, similar to a CA.

V. Security Proof and Analyses

To verify the equation $\hat{F}^s \stackrel{?}{=} \hat{R} \cdot \hat{E}^c$, we establish the relationship between $\hat{E}$ and $\hat{F}$. During the Sign phase:

$$\sigma_{agg} = \prod_{i=1}^{n} \sigma_i = \prod_{i=1}^{n} H^{a_i + b_i \cdot m_i} = H^{\sum(a_i + b_i m_i)}$$

And the aggregated public key is computed as $\tilde{K}_{agg} = \prod(\tilde{A}_i \cdot \tilde{B}_i^{m_i}) = \tilde{Y}^{\sum(a_i + b_i m_i)}$. We expand the value of $\hat{E}$ as follows:

$$\begin{aligned}
\hat{E} &= e(\sigma', \tilde{Y}) \cdot e(H', \tilde{K}_{agg})^{-1} \\
&= e((\sigma_{agg} \cdot H^t)^r, \tilde{Y}) \cdot e(H^r, \tilde{Y}^{\sum(a_i + b_i m_i)})^{-1} \\
&= e((H^{\sum(a_i + b_i m_i)})^r, \tilde{Y}) \cdot e(H^r, \tilde{Y})^t \cdot e(H^r, \tilde{Y})^{-\sum(a_i + b_i m_i)} \\
&= e(H^r, \tilde{Y})^{\sum(a_i + b_i m_i)} \cdot e(H', \tilde{Y})^t \cdot e(H^r, \tilde{Y})^{-\sum(a_i + b_i m_i)} \\
&= e(H', \tilde{Y})^t
\end{aligned}$$

Since $\hat{F} = e(H', \tilde{Y})$, we derive that $\hat{E} = \hat{F}^t$. Substituting $s = k + c \cdot t$ and $\hat{R} = \hat{F}^k$ into the right-hand side (RHS) of the verification equation: RHS $= \hat{R} \cdot \hat{E}^c = \hat{F}^k \cdot (\hat{F}^t)^c = \hat{F}^{k+c\cdot t} = \hat{F}^s =$ LHS. Thus, the first verification equation holds. Recall that during the Sign phase, $H' = H^r$, $\sigma' = (\sigma_{agg} \cdot H^t)^r$, and the aggregated public key is $\tilde{K}_{agg}$. We evaluate $\hat{E}$ by separating the target term $\hat{F}^t$ and a term Δ:

$$\begin{aligned}\hat{E} &= e(\sigma', \tilde{Y}) \cdot e(H', \tilde{K}_{agg})^{-1} \\ &= e((\sigma_{agg} \cdot H^t)^r, \tilde{Y}) \cdot e(H^r, \tilde{K}_{agg})^{-1} \\ &= \underbrace{\Big(e(\sigma_{agg}, \tilde{Y}) \cdot e(H, \tilde{K}_{agg})^{-1}\Big)^r}_{\Delta} \cdot \underbrace{e(H^r, \tilde{Y})^t}_{\hat{F}^t}\end{aligned}$$

We observe that $\hat{E}$ is composed of a validity term Δ and the target term $\hat{F}^t$. For the relation $\hat{E} = \hat{F}^t$ to hold, the term Δ must equal the identity element $1_{\mathbb{G}_T}$. Consequently, the equation $e(\sigma_{agg}, \tilde{Y}) = e(H, \tilde{K}_{agg})$ holds. This means that the validity of $\hat{F}^s = \hat{R} \cdot \hat{E}^c$ strictly implies that the signer possesses a valid PS signature, thereby confirming them as a legitimate user authorized by the issuers.

Given a TA-TWS signature $\sigma_x = (\prod_{i=1}^n T_{sk,i})^h \cdot X^x$ and the underlying tag relation $e(T_{sk,i}, \tilde{Y}) = \hat{T}_{vk,i}$, we substitute these directly into the verification equation:

$$\begin{aligned}e(\sigma_x, \tilde{Y}) &= e\left(\Big(\prod_{i=1}^n T_{sk,i}\Big)^h \cdot X^x, \tilde{Y}\right) \\ &= e\left(\prod_{i=1}^n T_{sk,i}, \tilde{Y}\right)^h \cdot e(X^x, \tilde{Y}) = \left(\prod_{i=1}^n e(T_{sk,i}, \tilde{Y})\right)^h \cdot \hat{Z}^x \\ &= \left(\prod_{i=1}^n \hat{T}_{vk,i}\right)^h \cdot \hat{Z}_x = \text{RHS}\end{aligned}$$

Therefore, the verification equation is satisfied, which establishes the completeness of TA-TWS.

A. Security Assumptions and Notation Explanation

The security analysis distinguishes computational hardness assumptions from the models used in the proofs. The $(2,1)$-DLOG assumption supports issuer authorization, the co-CDH assumption supports the binding of signatures to fresh requests, and the CBDH assumption controls the probability of querying the hidden hash input of an honestly generated tag. For each assumption, we specify the challenge distribution and the target output, and require that every PPT algorithm succeeds with only negligible probability.

(2,1)-DLOG Assumption. For uniformly random $\alpha \in \mathbb{Z}_q^*$, given a tuple $(q, X, X^\alpha, X^{\alpha^2}, \widetilde{Y}, \widetilde{Y}^\alpha, e)$, it is computationally infeasible for any PPT algorithm to recover α. Formally,

$$\Pr\Big[\mathcal{B}(q, X, X^\alpha, X^{\alpha^2}, \widetilde{Y}, \widetilde{Y}^\alpha, e) = \alpha : \alpha \leftarrow \mathbb{Z}_q^*\Big] \leq \text{negl}(\lambda)$$

co-CDH Assumption. Given a tuple $(g_1, g_1^a, g_2, g_2^a, g_2^b, e)$ for uniformly random $a, b \in \mathbb{Z}_q^*$, it is computationally infeasible for any PPT algorithm to compute $g_1^{ab} \in \mathbb{G}_1$. Formally,

$$\Pr\big[\mathcal{B}(g_1, g_1^a, g_2, g_2^a, g_2^b, e) = g_1^{ab} : a, b \leftarrow \mathbb{Z}_q^*\big] \leq \text{negl}(\lambda)$$

CBDH Assumption. For uniformly random $\rho, \theta, u \in \mathbb{Z}_q^*$, given a tuple $(g_1, g_1^\rho, g_1^\theta, g_1^u, g_2, g_2^\rho, g_2^\theta, g_2^u, e)$, it is computationally infeasible for any PPT algorithm to successfully compute the target value $e(g_1, g_2)^{\rho\theta u} \in \mathbb{G}_T$. Formally,

$$\Pr\left[\begin{array}{l}\mathcal{B}(g_1, g_1^\rho, g_1^\theta, g_1^u, g_2, g_2^\rho, g_2^\theta, g_2^u, e) \\ = e(g_1, g_2)^{\rho\theta u} : \rho, \theta, u \leftarrow \mathbb{Z}_q^*\end{array}\right] \leq \text{negl}(\lambda)$$

The algebraic group model (AGM) [45] requires the adversary to accompany each submitted group element with its algebraic representation, whereas the random-oracle model specifies the behavior of public hash functions. These models are distinct from computational hardness assumptions. In the bilinear setting, source-group representations are known scalar combinations of available inputs, while target-group representations may additionally include pairing terms.

Proof Notation. Let λ denote the security parameter and q the group order. Let n denote the number of issuers, and let U, L, M bound the numbers of users, generated tags, and tags in an aggregate, respectively. Let N denote the number of signatures in a batch and R an upper bound on the total number of batches. Participant counts and query bounds are polynomial in λ. We write $m_i = H_2(T_i)$ for the credential message and msg for the request message bound to a TA-TWS signature. The bounds Q_1, Q_2, Q_3 count the adversary's external calls to H_1, H_2, H_3, respectively, including repeated calls. The bound Q_4 counts independent H_4 challenge samples within one execution, including answers to fresh inputs and programming samples used in signing simulation. Let q_s bound the number of signing queries. The records Q_c, Q_w, Q_s contain, respectively, individual tags for which credentials have been issued, individual tags whose complete witnesses have been returned to the adversary, and pairs $(\mathcal{T}, msg)$ for which TA-TWS signatures have been successfully returned. Internal issuance updates Q_c, whereas internal witness use does not update Q_w.

B. The Forking Lemma

The Forking Lemma [46] is an extraction technique in the random-oracle model. Suppose an adversary makes at most a polynomial number of hash queries and outputs an accepting proof with non-negligible probability. A reduction can rewind the adversary to the hash query that fixes the challenge, reuse the same random tape and preceding oracle answers, and provide an independent challenge at that query. If both runs accept with the same commitment but different challenges, and the verification equations have the required extraction property, their responses can be combined to extract the value underlying the proof. The lemma bounds the probability of obtaining such a pair in terms of the original success probability, the number of hash queries, and the challenge-space size.

C. Security Proofs

Before presenting the security proofs, we replace the base-point derivation function $f(rsk, \widetilde{PK}_U, st)$ with a random function that returns the same H for repeated inputs and

independently samples $H \leftarrow \mathbb{G}^*$ for each new input. We instantiate f with a secure PRF whose outputs are computationally indistinguishable from those of this random function. Therefore, this replacement changes the adversary's success probability by at most $\Delta_f \leq \mathrm{negl}(\lambda)$.

Shared Proof Conventions. For conciseness, we summarize the conventions used throughout the security proofs to avoid repetition. Long-term secret-key exponents and credential-base exponents are nonzero, and their corresponding public keys and bases are nonidentity elements. The simulation resamples any such public key or base if it equals the identity, while private coefficients in affine decompositions may still be zero. Other scalars follow the sampling domains specified in the construction. Rare zero values of tag randomness or temporary secrets, accidental hash-input or independently sampled base collisions, and bounded-resampling errors are absorbed into negligible terms. Issuance records are indexed by the actual base, and all simulations follow the one-time issuance and caching rules in Fig. 1. These conventions apply to honest generation, while adversarial signature elements remain subject to verification checks. The challenge-space sizes of H_3 and H_4 are denoted by κ_3 and κ_4, respectively, both equal to q for uniform outputs in $\mathbb{Z}_q$. We use H_{blind} to denote the credential-blinding hash, whose outputs lie in $\mathbb{Z}_q^*$. Bounds appearing in denominators are assumed to be at least one. Algebraic representations and coefficient computations have polynomial size and running time.

Lemma 1. *In the random-oracle game following the aforementioned base-point replacement under our threat model, let μ_1 denote the probability that the adversary queries H_1 at least once on the complete input $(X_t, \widetilde{Y}_t, e(PK_R^t, \widetilde{PK}_U))$ corresponding to a genuine tag. If the CBDH assumption holds, then μ_1 is negligible.*

Proof. Upon receiving a CBDH challenge, the reduction $\mathcal{B}_1$ uniformly guesses a target user U^*, a target tag-generation index $\ell^* \in \{1, \ldots, L\}$, and the H_1 query position $k^* \in \{1, \ldots, Q_1\}$ at which a genuine hidden input is first submitted. It sets $X = g_1$, $\widetilde{Y} = g_2$, $PK_R = g_1^\rho$, and $\widetilde{PK}_{U^*} = g_2^u$. The keys of all other users and all issuers are generated according to the real algorithms. When the ℓ^*-th tag-generation request occurs, the reduction $\mathcal{B}_1$ aborts if the requesting user is not U^*. Otherwise, it implicitly sets the target tag's randomness to $t^* = \theta$ without knowing its value, and uses it to set $X_{t^*} = g_1^\theta$ and $\widetilde{Y}_{t^*} = g_2^\theta$. The target tag's hidden third component is therefore $e(PK_R^{t^*}, \widetilde{PK}_{U^*}) = e(g_1^\rho, g_2^u)^\theta = e(g_1, g_2)^{\rho\theta u}$, which is precisely the target value of the CBDH challenge.

Query Phase. The reduction $\mathcal{B}_1$ answers hash queries according to the rules in Fig. 1. Repeated inputs receive the same responses, whereas new inputs are answered with independent uniformly sampled values. Credential base points are generated using the shared random table defined in the preceding base-point replacement game, ensuring that identical derivation inputs yield identical base points.

When generating the target tag, $\mathcal{B}_1$ samples $v^* \leftarrow \mathbb{Z}_q$ and computes $T_{sk,*} = (g_1^u)^{v^*}$ and $\widehat{T}_{vk,*} = e(T_{sk,*}, g_2)$. For other tags belonging to the same target user, the reduction $\mathcal{B}_1$ knows their generation randomness t_j and can therefore compute $v_j = H_1\Big(g_1^{t_j}, g_2^{t_j}, e(g_1^\rho, g_2^u)^{t_j}\Big)$ and $T_{sk,j} = (g_1^u)^{v_j}$. Tags belonging to non-target users are generated according to the real algorithm. $\mathcal{B}_1$ knows all issuer secret keys and can compute the blinding hash input $(\widetilde{PK}_{U^*})^{a_i}$. It can therefore answer credential, complete witness, and TA-TWS signing queries according to the real algorithms while maintaining the corresponding query records and one-time issuance cache.

Indistinguishability. Before the adversary $\mathcal{A}$ first queries the genuine hidden input of the target tag, the corresponding H_1 value in the real protocol is uniformly random and has not yet been returned to $\mathcal{A}$. Thus, the reduction $\mathcal{B}_1$ samples a uniform v^* in advance and uses it to compute the temporary secret and its pairing-based public key. This yields the same joint distribution as the real protocol up to this point. Other permitted queries are answered according to their real distributions. Under the shared conventions, the mapping $v \mapsto usk_U v$ is bijective. When the complete hash inputs of different tags are distinct, their hash outputs are independent. Therefore, the temporary secrets of different tags can be sampled as independent uniform group elements while preserving the consistency of their pairing-based public keys.

Consider an analysis game that terminates when the adversary first submits a genuine hidden input. Before this event, the tags, credentials, witnesses, signatures, and hash responses observed by the adversary have the same joint distribution as in the real game. Consequently, the difference between the two game views is bounded by the probability μ_1 of this stopping event, plus negligible errors arising from premature random-prefix matches, zero values, and input collisions.

Reduction and Probability Analysis. When the adversary makes its k^*-th H_1 query, the reduction $\mathcal{B}_1$ examines the first two components of the query input. If they equal (g_1^θ, g_2^θ), the reduction $\mathcal{B}_1$ outputs the third component and terminates. Otherwise, the reduction $\mathcal{B}_1$ fails. When all three guesses are correct, this query is exactly the first query containing the genuine hidden input of the target tag. Its third component must therefore equal $e(g_1, g_2)^{\rho\theta u}$. Hence, the reduction $\mathcal{B}_1$ does not need to verify the correctness of the third component.

If the adversary queries a genuine hidden input at least once, the probability that $\mathcal{B}_1$ correctly guesses the target user, target tag-generation position, and first matching query position is at least $1/(ULQ_1)$. When all three guesses are correct and no simulation failure occurs, the reduction $\mathcal{B}_1$ outputs the third component as a valid CBDH solution. Therefore, $\mathrm{Adv}_{\mathcal{B}_1}^{\mathrm{CBDH}} \geq \frac{\mu_1}{ULQ_1} - \mathrm{negl}(\lambda)$. Since U, L, and Q_1 are polynomially bounded in the security parameter, a non-negligible μ_1 would allow $\mathcal{B}_1$ to solve the CBDH problem with non-negligible probability, contradicting the CBDH assumption. Thus, μ_1 is negligible. The reduction $\mathcal{B}_1$ runs the adversary only once, with additional overhead consisting of polynomially many group operations and record-maintenance steps. □

Theorem 1. *In the AGM and ROM, if the $(2,1)$-DLOG, co-CDH, and CBDH assumptions hold and the base-point derivation function f satisfies PRF security, then TA-TWS achieves the EUF-CTMA security defined in Section IV-C.*

By the definitions and update rules of Q_c, Q_w, Q_s and the winning condition in Section IV-C, a successful forgery

must pass verification, contain at least one tag in $\mathcal{T}^*$ outside Q_w, and satisfy $(\mathcal{T}^*, msg^*) \notin Q_s$. The experiment permits adaptive queries for different tags of the same user and their credentials, witnesses, and TA-TWS signatures, including tags outside the final list. Complete witnesses for some final components may already have been returned, but at least one must remain undisclosed. Using the actual issuance records at the end of the experiment, we partition successful forgeries into two mutually exclusive and exhaustive events:

- **Type-I forgery:** At least one final tag is outside Q_c; the adversary attempts to bypass issuer authorization.
- **Type-II forgery:** Every final tag is in Q_c; the adversary attempts to sign a fresh list-message pair while at least one complete witness remains protected.

By PRF security, replacing base-point derivation with a consistent random function changes the success probability by at most the negligible distinguishing loss Δ_f. Let ϵ_1, ϵ_2 denote the probabilities of these two events in the modified game. The actual forgery probability ϵ satisfies $\epsilon \leq \Delta_f + \epsilon_1 + \epsilon_2$. The following reductions show that both event probabilities are negligible, thereby proving Theorem 1. All reductions follow the oracle interfaces and record-update rules in Fig. 1 of Section IV-C. We only describe their simulation.

Lemma 2. *In the AGM and ROM, the* $(2,1)$*-DLOG assumption implies the probability* ϵ_1 *of a Type-I forgery is negligible. Otherwise, a PPT reduction* $\mathcal{B}_2$ *solves* $(2,1)$*-DLOG.*

Proof. The reduction $\mathcal{B}_2$ receives the $(2,1)$-DLOG challenge $(X, X^\alpha, X^{\alpha^2}, \widetilde{Y}, \widetilde{Y}^\alpha)$. Among n honestly initialized issuers, $\mathcal{B}_2$ uniformly guesses a target $i^* \in \{1, \ldots, n\}$. It samples private coefficients $a_0, a_1, b_0, b_1 \leftarrow \mathbb{Z}_q$ and publishes $\widetilde{A}_{i^*} = \widetilde{Y}^{a_0}(\widetilde{Y}^\alpha)^{a_1}$, $\widetilde{B}_{i^*} = \widetilde{Y}^{b_0}(\widetilde{Y}^\alpha)^{b_1}$. This implicitly sets $a = a_0 + a_1\alpha$ and $b = b_0 + b_1\alpha$. The reduction $\mathcal{B}_2$ knows the coefficients, but neither α nor the actual values of a, b. It honestly generates and knows the secrets of all other issuers, all users, and the RA. The challenge points $X^\alpha, X^{\alpha^2}, \widetilde{Y}^\alpha$ are available only to the reduction $\mathcal{B}_2$; they are not additional protocol parameters supplied to the adversary. The adversary receives only the prescribed TA-TWS parameters, public keys, and oracle answers, with the distribution established below.

Query Simulation. For base-point derivation, when a user-state input first requires a base, $\mathcal{B}_2$ samples private coefficients $z_{\beta,0}, z_{\beta,1} \leftarrow \mathbb{Z}_q$ and computes $H_\beta = X^{z_{\beta,0}}(X^\alpha)^{z_{\beta,1}}$. The index β identifies derivation inputs. The reduction $\mathcal{B}_2$ knows the affine coefficients but not the actual exponent $z_\beta = z_{\beta,0} + z_{\beta,1}\alpha$, and reuses the same base for repeated inputs.

For $\mathcal{O}^{\mathrm{Tag}}$ queries, $\mathcal{B}_2$ chooses fresh tag randomness for the specified user and issuer, computes the tag and $T_{sk,i}, \widehat{T}_{vk,i}$ honestly, and saves the context. Knowing the user and RA secrets, it can compute the hidden hash input directly.

For $\mathcal{O}^{\mathrm{Cred}}$ queries to the target issuer, let $m = H_2(T)$. The unblinded credential satisfies $\sigma_i = H_\beta^{a+bm}$, with exponent

$$\begin{aligned} z_\beta(a+bm) =& z_{\beta,0}(a_0 + b_0 m) + \big[z_{\beta,0}(a_1 + b_1 m) \\ &+ z_{\beta,1}(a_0 + b_0 m)\big]\alpha + z_{\beta,1}(a_1 + b_1 m)\alpha^2. \end{aligned}$$

Even if m is chosen after H_β is revealed, the three first-source-group challenge points suffice to compute

$$\begin{aligned} \sigma_i =& X^{z_{\beta,0}(a_0+b_0 m)} \cdot (X^\alpha)^{z_{\beta,0}(a_1+b_1 m)+z_{\beta,1}(a_0+b_0 m)} \\ &\cdot (X^{\alpha^2})^{z_{\beta,1}(a_1+b_1 m)} = H_\beta^{a+bm}. \end{aligned}$$

Once m is known, all exponent coefficients are computable. Thus, the simulation permits revealing a base before adaptively selecting a message, without assuming that the base and message are independent. Non-target issuer credentials use known secrets. For each issuer and actual base, the first successful issuance adds the tag to Q_c; repeated requests for the same tag return the cached credential, while a different message is rejected. The target issuer's blinding factor can also be generated. Although a is unknown, the user secret is known, so the reduction $\mathcal{B}_2$ computes $\widetilde{A}_{i^*}^{usk_U}$ and queries H_{blind} to produce the blinded partial credential.

For $\mathcal{O}^{\mathrm{Wit}}$ queries, the reduction $\mathcal{B}_2$ returns $(H_\beta, \sigma_i, T_{sk,i})$, using a previously issued or internally issued credential, and updates Q_c, Q_w as specified in Section IV-C. For $\mathcal{O}^{\mathrm{Sign}}$ queries on valid aggregate lists, it obtains component credentials internally, generates both proofs honestly, and updates Q_c, Q_s. Internal witness use does not update Q_w. Hash answers are consistent and uniformly sampled for new inputs. When an H_3 input first appears, reduction $\mathcal{B}_2$ saves its algebraic representation before returning the challenge. If an honest signature reuses that input, its original aggregate-public-key representation is also saved. The adversary may continue making valid queries afterward.

Simulation Distribution. For fixed actual α, uniformly sampling (w_0, w_1) makes $w = w_0 + w_1\alpha$ uniform. Conditioned on w, every slope w_1 corresponds to exactly one intercept $w_0 = w - w_1\alpha$, so the slope remains uniform. Conditioning on $w \neq 0$ preserves this property and makes the actual secret uniform in $\mathbb{Z}_q^*$. The observed group encodings, credentials, hash answers, and TA-TWS signatures depend only on actual secrets and real protocol randomness, not on the private affine decompositions used by $\mathcal{B}_2$. Even learning actual secrets from encodings does not reveal these slopes. Real protocol randomness here excludes the additional coefficients used solely for affine decomposition.

Forgery Output. The adversary outputs an accepting Type-I forgery $(\mathcal{T}^*, msg^*, \Sigma^*)$. The reduction $\mathcal{B}_2$ fails if its guessed issuer does not correspond to an unissued final component. Every successful Type-I forgery contains at least one such component, so the guess succeeds with probability at least $1/n$. On a correct guess, $T_{i^*}^*$ is a generated tag of issuer i^* in the final list whose credential has not been issued, i.e., $T_{i^*}^* \notin Q_c$. Initialization guesses only the issuer; the adversary's final output determines the target tag. Let $m^* = H_2(T_{i^*}^*)$. Excluding H_2 collisions between distinct tags, m^* never occurs in the target issuer's issuance records during the entire execution; otherwise, the target tag would be in Q_c.

This condition uses the complete records at final output, including issuance performed internally by signing queries. Issuing the target tag after the H_3 challenge but before final output removes it from the unauthorized components and may invalidate the guess. The output remains Type-I if

another final component is still unissued. From the saved AGM representations at the first occurrence of the final H_3 input, the reduction $\mathcal{B}_2$ derives formal exponents $H' = X^P, \sigma' = X^Q, \widetilde{K}_{\text{agg}} = \widetilde{Y}^{K_0}, \widehat{R} = \widehat{Z}^{R_0}$. The coefficients of P, Q, K_0, R_0 are computable from the representations. These representations are fixed before the input's first challenge c is returned. The adversary may subsequently choose s, but later credentials cannot change the saved representations. The correct aggregate public key for the final list has formal exponent

$$a + bm^* + \Delta, \qquad \Delta = \sum_{\substack{j \in I^* \\ j \neq i^*}} (a_j + b_j H_2(T_j^*)),$$

where I^* is the final issuer-index set. The reduction $\mathcal{B}_2$ knows the other issuer secrets and computes Δ, using each issuer's own tag message $H_2(T_j^*)$. The exponent K_0 comes from the first query's representation, whereas $a + bm^* + \Delta$ comes from the final list. They represent the same actual group element but need not be identical formal polynomials. Their difference vanishes at the actual secrets but may have nonzero coefficients. We make the same distinction for the public-key and authorization relations: a formally nonzero relation that vanishes at the actual secrets yields an extraction attempt; formal identities lead to a contradiction with nonissuance.

Security Reduction We analyze an accepting Type-I forgery in two cases according to the coefficients of the polynomial relations obtained from its AGM representations. These relations evaluate to zero at the actual secret values, but their coefficients need not all be zero. In Case 1, at least one relation has a nonzero coefficient. The reduction $\mathcal{B}_2$ uses this relation to obtain a polynomial equation in the unknown exponent α and recover the $(2,1)$-DLOG solution, except with negligible probability. In Case 2, all coefficients of every checked relation are zero. We show that, apart from challenge-guessing events, this case requires a credential for the unissued target message to be expressible using the available source-group inputs, contradicting the target message's nonissuance.

Case 1. All coefficients and exponent arithmetic are over $\mathbb{Z}_q$. Throughout execution, first-source-group exponents are known scalar linear combinations of $\{1, z_\beta, z_\beta(a + bm_\beta)\}$, where the third term is available only if the target issuer has issued a credential at that base. Indeed, the RA public key, tag random points, and $T_{sk,i}$ have exponents known to the reduction $\mathcal{B}_2$. Non-target credentials are known multiples of z_β; target blinded credentials are known multiples of $z_\beta(a + bm_\beta)$. Honest H', σ' are linear combinations of existing bases and credentials, while σ_x has a known exponent. A visible blinding-hash input has second-source exponent $usk_U \cdot a$, also in $\text{span}\{1, a, b\}$. Public pairings and target-group inputs retain their AGM representations, and later inputs extend the basis by the same rules.

After substituting affine expressions in α, exponent polynomials have degree at most two in the first source group, one in the second, and three in the target group. The protocol provides no map from the target group back to a source group, and scalar hashes do not produce source-group elements with additional unknown discrete logarithms. By bilinearity,

$$\widehat{E} = e(X^Q, \widetilde{Y}) e(X^P, \widetilde{Y}^{K_0})^{-1} = \widehat{Z}^{Q - PK_0},$$
$$\widehat{Z} = e(X, \widetilde{Y}), \quad \widehat{F} = e(X^P, \widetilde{Y}) = \widehat{Z}^P.$$

The exponent R_0 is derived from the AGM representation of $\widehat{R}$ through multiplication, exponentiation, and pairing; we do not assume the adversary generated it honestly as $\widehat{F}^k$. Acceptance of $\widehat{F}^s = \widehat{R}\widehat{E}^c$ therefore implies $sP - R_0 - c(Q - PK_0) = 0$ at the actual secrets. For an accepting record, the reduction $\mathcal{B}_2$ checks the following relations in a predetermined order:

1) The difference between K_0 and $a + bm^* + \Delta$, followed, in invocation order, by the differences between the original aggregate-key exponents of honest signatures reusing this hash input and the final correct exponent.
2) The exponent P, if the actual element $H' = 1$.
3) The authorization residual $sP - R_0 - c(Q - PK_0)$.

These relations vanish at the actual secrets. If any is formally nonzero, the reduction $\mathcal{B}_2$ selects the first such relation $\mathcal{L}$ and substitutes its affine expressions to obtain

$$\ell(T) = \mathcal{L}\big(a_0 + a_1 T, b_0 + b_1 T, z_{1,0} + z_{1,1} T, \dots\big).$$

Here T is only a formal root-finding variable. The polynomial ℓ is computable from known coefficients, has degree at most three, and satisfies $\ell(\alpha) = 0$. Consider the possibility that substitution makes ℓ identically zero. Conditioned on the complete actual execution, the coefficients of $\mathcal{L}$ are fixed, the private slopes remain independent and uniform, and relation selection does not depend on them. If $d = \deg \mathcal{L}$, the coefficient of T^d in ℓ is the nonzero highest-degree homogeneous part of $\mathcal{L}$ evaluated at the slope vector. If $\ell \equiv 0$, this coefficient must vanish. The finite-field polynomial root bound gives

$$\Pr[\ell \equiv 0 \mid \mathcal{L} \not\equiv 0] \leq d/q \leq 3/q.$$

Otherwise, $\mathcal{B}_2$ finds at most three roots of ℓ and compares their powers of $\widetilde{Y}$ with $\widetilde{Y}^\alpha$ to identify α. Root finding for constant-degree finite-field polynomials takes randomized polynomial time; a polynomial retry limit makes failure negligible. Thus, reduction $\mathcal{B}_2$ directly solves $(2,1)$-DLOG.

Case 2. Suppose all checked relations are formal identities. Honest signatures can satisfy these identities; this branch does not exclude all accepting identity records. Instead, it shows that an unissued target message cannot yield a new credential expressed as a known scalar linear combination of available source-group inputs.

First, the forgery cannot reuse an H_3 input used internally by an honest signature. If that signature's original public-key exponent equals the final exponent formally, its coefficients of a, b are $1, m^*$, so it already contains the target credential message. Excluding H_2 collisions, the target tag is identical and belongs to Q_c, contradicting nonissuance. If the two formal exponents differ while representing the same element, the previous branch yields a nonzero relation. This check also covers inputs first queried by the adversary and later reused internally by an honest signature.

Next, consider $H' = 1$. If P is formally nonzero, the previous branch extracts the DLOG solution. If $P \equiv 0$, then $\widehat{F} = 1$. If also $\widehat{E} = 1$, acceptance requires $\sigma' = \widehat{R} = 1$ and

the all-identity check rejects. If $\widehat{E} \neq 1$, at most one challenge for the fixed hash input permits acceptance; this contributes to the challenge-guessing loss.

Now suppose $P \not\equiv 0$. If $Q - PK_0$ is not a scalar multiple of P, at most one challenge c permits a response s making $sP = R_0 + c(Q - PK_0)$ a formal identity. Indeed, if distinct challenges c, c' admit responses s, s', subtracting the two identities yields $Q - PK_0 = (s - s')(c - c')^{-1}P$, contradicting nonproportionality. This comparison establishes uniqueness. A union bound over at most Q_3 external first queries and one possible new final-verification query bounds these events by $(Q_3 + 1)/\kappa_3$. The bound permits choosing responses and continuing queries after receiving a challenge, since the relevant representations were fixed beforehand.

In the remaining case, $Q - PK_0 = t^*P$ for a scalar t^* computable from coefficient ratios. The reduction $\mathcal{B}_2$ divides the coefficient of any nonzero monomial in P into the matching coefficient of $Q - PK_0$ and checks that all other coefficients have the same ratio. The scalar t^* is the credential-randomization offset, not a user secret or tag-generation exponent. From a single signature record, reduction $\mathcal{B}_2$ extracts

$$\sigma^*_{\mathrm{agg}} = \sigma'(H')^{-t^*}, e(\sigma^*_{\mathrm{agg}}, \widetilde{Y}) = e(H', \widetilde{K}_{\mathrm{agg}}).$$

Removing the other issuers' contributions using their respective tag messages gives $\sigma^*_{i^*} = \sigma^*_{\mathrm{agg}}(H')^{-\Delta} = (H')^{a+bm^*}$. This is a PS-form credential relation. We next check whether available source-group inputs can express it. Suppose a nonzero formal polynomial P makes $P(a + bm^*)$ belong to the available exponent space. Write

$$P = p_0 + \sum_{\beta} \mu_\beta z_\beta + \sum_{\beta:\,\text{issued}} \lambda_\beta z_\beta (a + bm_\beta).$$

The space permits known scalar linear combinations, not arbitrary multiplication by unknown secrets. Comparing coefficients in $P(a + bm^*)$ with those available gives:

1) The coefficient of $z_\beta a^2$ is λ_β. No such monomial is available, so all $\lambda_\beta = 0$, leaving $P = p_0 + \sum_\beta \mu_\beta z_\beta$.
2) The coefficient of a without a base variable is p_0. No standalone a term is available, so $p_0 = 0$.
3) For a base with no target-issuer credential, no $z_\beta a$ term is available, so $\mu_\beta = 0$. For an issued base, the one-time rule supplies only one target-issuer message m_β. Its $z_\beta a, z_\beta b$ coefficients have ratio $1 : m_\beta$, whereas the new credential requires $1 : m^*$. Hence $\mu_\beta(m^* - m_\beta) = 0$.

The target message is unissued throughout the entire execution. Excluding H_2 collisions, $m^* \neq m_\beta$ at every issued base. Thus, all $\mu_\beta = 0$, yielding $P = 0$, a contradiction. Yet the extracted relation requires $P(a + bm^*) = Q - (t^* + \Delta)P$ to belong to the same source-group exponent space. Therefore, a formally identical new-message credential cannot be constructed. Later bases and credentials cannot alter saved representations; even enlarging the space with all later inputs leaves the contradiction intact because the target message remains unissued.

The case $\widehat{E} = 1$ corresponds to $t^* = 0$ and is included. An unqueried final H_3 input cannot automatically be classified as challenge guessing, since $\widehat{E} = 1$ may make verification independent of the challenge. Such cases are also handled by nonzero-relation extraction or the source-space contradiction.

Probability and Running Time. The reduction $\mathcal{B}_2$ satisfies

$$\mathrm{Adv}^{(2,1)\text{-DLOG}}_{\mathcal{B}_2} \geq \frac{\epsilon_1}{n} - \frac{Q_3 + 1}{\kappa_3} - \frac{3}{q} - \mathrm{negl}(\lambda).$$

ϵ_1/n accounts for forgery success and the issuer guess. The loss $(Q_3{+}1)/\kappa_3$ covers the nonproportional and corresponding identity-element cases with at most one accepting challenge per fixed input. The loss $3/q$ bounds an identically zero polynomial after affine substitution. The remaining negligible term covers H_2 collisions, accidental base collisions across states, bounded resampling, and root-finding failure. There are at most $Q_2 + L$ distinct external and tag-related internal H_2 inputs, giving collision probability $O((Q_2 + L)^2/q)$.

These losses remove exceptional executions that do not guarantee extraction. A nonzero relation yields α; otherwise, outside the exceptions, successful formal-identity forgery yields an algebraic contradiction. Polynomial n, Q_3 and non-negligible ϵ_1 would contradict $(2, 1)$-DLOG. The PRF loss Δ_f is accounted for in the theorem. The reduction $\mathcal{B}_2$ runs the adversary once, with polynomially many group operations and coefficient checks and one degree-at-most-three root-finding computation. Hence ϵ_1 is negligible. □

Lemma 3. *In the AGM and random-oracle model, the CBDH and co-CDH assumptions imply that the probability ϵ_2 of a Type-II forgery is negligible. Otherwise, one can solve CBDH via $\mathcal{B}_1$ or co-CDH via the PPT reduction $\mathcal{B}_3$.*

Proof. By the CBDH reduction $\mathcal{B}_1$ in Lemma 1, the probability μ_1 of querying a genuine tag's complete hidden input is negligible. Before the first such query, the temporary secrets $T_{sk,i}$ can equivalently be generated as independent uniform group elements, preserving $\widehat{T}_{vk,i} = e(T_{sk,i}, \widetilde{Y})$. Replacing tag-secret generation by this sampling changes the view by at most μ_1 plus negligible statistical error.

System Initialization. $\mathcal{B}_3$ receives the co-CDH challenge $(g_1, g_1^a, g_2, g_2^a, g_2^b)$ and aims to compute g_1^{ab}. It sets $X = g_1, \widetilde{Y} = g_2$, honestly generates all issuer, user, and RA keys, and knows their secrets. It uniformly guesses a tag-generation index $\ell^* \in \{1, \ldots, L\}$ and, when that tag is generated, sets $\widehat{T}_{vk,\ell^*} = e(g_1^a, g_2^b)$. The corresponding unknown temporary secret is $T_{sk,\ell^*} = g_1^{ab}$. For other tags, it chooses known random scalars ω_i and sets $T_{sk,i} = X^{\omega_i}$ with $\widehat{T}_{vk,i} = e(T_{sk,i}, \widetilde{Y})$. Nonzero challenge exponents a, b make the target secret uniform over nonidentity elements, differing from uniform group sampling only by negligible identity probability. Only the target tag's temporary secret contains the unknown value; user long-term keys do not. Thus, temporary secrets for other tags of the same target user remain known and their witness queries can be answered. Knowing user secrets and tag-generation randomness, the reduction $\mathcal{B}_3$ identifies genuine hidden inputs and aborts at their first adversarial query; it need not know the hash scalar corresponding to the special temporary secret.

Query Simulation. For a complete-witness query on a non-target tag, the reduction $\mathcal{B}_3$ generates or reuses a credential under the one-time rule, returns the known complete witness, and updates Q_c, Q_w as specified in Section IV-C. It aborts on

a target-tag complete-witness query. The probability analysis retains successful executions in which the final list contains the guessed tag and its complete witness remains undisclosed; this abort is therefore handled together with the target guess.

For TA-TWS signing queries, all issuer secrets are known, so authorization credentials and their proof are generated honestly. If the list excludes the target tag, message binding is also generated honestly. Otherwise, the reduction $\mathcal{B}_3$ uniformly samples $\sigma_x \in \mathbb{G}$ and a challenge h, and computes

$$\widehat{Z}_x = e(\sigma_x, \widetilde{Y}) \left(\prod_{i \in I} \widehat{T}_{vk,i} \right)^{-h}.$$

It resamples if $\widehat{Z}_x = 1$ or the corresponding H_4 input already has an answer. Otherwise, it records answer h at the fresh input $H_4(msg, \widehat{Z}_x, (\widehat{T}_{vk,i})_{i \in I})$, returns the signature, and updates Q_c, Q_s. Existing hash answers are never overwritten, and internal witness use does not update Q_w.

Simulation Distribution. For fixed h, the map $\sigma_x \mapsto \widehat{Z}_x$ is bijective. Rejecting the identity makes $\widehat{Z}_x$ uniform over $\mathbb{G}_t^*$, with a unique nonzero x satisfying $\widehat{Z}_x = \widehat{Z}^x$. Injectivity of the pairing map gives $\sigma_x = \left(\prod_i T_{sk,i}\right)^h X^x$, the honest message-binding response. At an unanswered hash input, the simulated commitment, challenge, and response thus have the honest joint distribution. For at most q_s signing queries, conflicts with existing inputs and bounded resampling contribute statistical difference $O\left(\frac{q_s(Q_4+q_s)}{q-1}\right) + \mathrm{negl}(\lambda)$. The count Q_4 includes challenge samples used to program fresh signing inputs.

Forgery Output. $\mathcal{B}_3$ retains only valid Type-II forgeries containing the guessed tag $T_{\ell^*} \notin Q_w$. Every successful Type-II forgery has at least one undisclosed complete witness, so the target guess succeeds with probability at least $1/L$.

Temporary public keys collide with negligible probability. Excluding them, the canonically encoded temporary-public-key list uniquely identifies the complete tag list. An adversary may query H_4 using a public key of a not-yet-generated tag; a later tag has that key with probability at most $O(LMQ_4/(q-1))$. Excluding this event, all final tags exist before the relevant fork point, and rewinding cannot change their identities.Security

Security Reduction. First, if $\prod_i \widehat{T}_{vk,i} = 1$, injectivity of the pairing map implies that the temporary-secret product is also the identity. The reduction $\mathcal{B}_3$ directly computes

$$g_1^{ab} = T_{sk,\ell^*} = \left(\prod_{i:\, T_i \neq T_{\ell^*}} T_{sk,i} \right)^{-1}.$$

All factors except the target secret are known, so this branch requires no prior H_4 query.

Now suppose the temporary-public-key product is nonidentity. If the adversary never queried the final signature's H_4 input, at most one random challenge accepts the fixed output, giving probability at most $1/\kappa_4$. Nor can this input have been recorded by a signing query: the same request message and canonical public-key list identify the same list-message pair, placing $(\mathcal{T}^*, msg^*)$ in Q_s and contradicting freshness.

In the remaining successful executions, the reduction $\mathcal{B}_3$ locates the external query

$$H_4(msg^*, \widehat{Z}_x^*, (\widehat{T}_{vk,i})_{i \in I^*}).$$

By the General Forking Lemma [46], it keeps the adversary's random tape and the answers before this query fixed, and resamples challenges from this query onward. With the lower bound given below, it obtains two accepting records with the same hash input and distinct challenges h^*, h', satisfying

$$e(\sigma_x^*, \widetilde{Y}) = \widehat{Z}_x^* \left(\prod_i \widehat{T}_{vk,i} \right)^{h^*},\ e(\sigma_x', \widetilde{Y}) = \widehat{Z}_x^* \left(\prod_i \widehat{T}_{vk,i} \right)^{h'}$$

Dividing the equations and using $h^* \neq h'$ and pairing-map injectivity gives $\left(\frac{\sigma_x^*}{\sigma_x'}\right)^{(h^*-h')^{-1}} = \prod_{i \in I^*} T_{sk,i}$. Removing the other known temporary secrets yields

$$g_1^{ab} = \left(\frac{\sigma_x^*}{\sigma_x'}\right)^{(h^*-h')^{-1}} \left(\prod_{\substack{i \in I^* \\ T_i \neq T_{\ell^*}}} T_{sk,i} \right)^{-1}.$$

Both accepting records require that the same target tag's complete witness remain undisclosed; its identity is fixed before the fork. Other random functions remain consistent, while Q_c, Q_w, Q_s and the one-time issuance cache are restored from the common prefix. Since all issuer secrets are known, the reduction $\mathcal{B}_3$ answers subsequent issuance queries separately in each branch, each obeying its own one-time rule. No external one-time issuance interface is rewound. Thus, forking is compatible with issuance state. First-run success does not guarantee second-run success; the General Forking Lemma accounts for this loss.

Probability Analysis. Let δ denote the probability of a correct target guess and successful direct extraction, and γ the probability of a correct guess, an undisclosed target witness, and first-run success with an identifiable fork point. Then

$$\delta + \gamma \geq \frac{\epsilon_2 - \mu_1}{L} - \frac{1}{\kappa_4} - \mathrm{negl}(\lambda) - O\left(\frac{q_s(Q_4+q_s) + LMQ_4}{q-1}\right).$$

The first term removes genuine-hidden-input hits and accounts for the target guess. The statistical term covers signing-simulation conflicts and premature matches of future tag public keys; $1/\kappa_4$ covers an unqueried final hash input.

Direct extraction and fork extraction are mutually exclusive branches. Combining them with the General Forking Lemma gives reduction $\mathcal{B}_3$ satisfying

$$\mathrm{Adv}_{\mathcal{B}_3}^{\text{co-CDH}} \geq \delta + \max\left\{0, \gamma\left(\frac{\gamma}{Q_4} - \frac{1}{\kappa_4}\right)\right\}.$$

The terms are the direct-extraction probability and the lower bound for two accepting records with distinct challenges. The maximum with zero accounts for nonnegativity when the forking expression is negative. By the CBDH reduction $\mathcal{B}_1$ in Lemma 1, μ_1 is negligible. Polynomial participant and query bounds and exponentially large κ_4 make the other errors negligible. Non-negligible ϵ_2 would therefore imply non-negligible $\delta + \gamma$, so at least one extraction branch solves co-CDH with non-negligible probability. The reduction $\mathcal{B}_3$

runs the adversary at most twice and imposes polynomial resampling limits, giving at most two adversary executions plus polynomial simulation overhead. Hence ϵ_2 is negligible. □

Theorem 2. *In the programmable ROM, for a valid tag list and request message, there exists a PPT simulator using only public inputs whose joint signature and visible random-oracle view has negligible statistical distance from the real view.*

Proof. We construct a simulator $\mathcal{S}$ that uses the programmable random oracles H_3, H_4 without access to the witness w. It reconstructs the public values $\widetilde{K}_{\mathrm{agg}}$ and $\widehat{T}_{vk,i}$ and then generates $\Sigma_{\mathrm{sim}} = (H', \sigma', \widehat{R}, s, \widehat{Z}_x, \sigma_x)$ in two steps.

Step 1: Sample $H' \leftarrow \mathbb{G}^*$ and $\sigma' \leftarrow \mathbb{G}$, and compute

$$\widehat{F} = e(H', \widetilde{Y}), \widehat{E} = e(\sigma', \widetilde{Y}) e(H', \widetilde{K}_{\mathrm{agg}})^{-1}.$$

Resample σ' if $\widehat{E} = 1$. Sample a uniform challenge c and $s \leftarrow \mathbb{Z}_q$, and set $\widehat{R} = \widehat{F}^s \widehat{E}^{-c}$. Resample (c, s) if $\widehat{R} = 1$. If the corresponding H_3 input already has an answer, resample this step. Otherwise, record c at the input $(msg, \widehat{R}, \widetilde{K}_{\mathrm{agg}}, H', \sigma')$.

In the real algorithm, $r \in \mathbb{Z}_q^*$ makes H^r uniform over $\mathbb{G}^*$. Conditioned on H', nonzero t makes $\sigma' = (H')^{\sum_i (a_i + b_i m_i) + t}$ uniform over the group excluding the unique element that yields $\widehat{E} = 1$. The simulator has the same distribution. For fixed t, c, nonzero k makes $s = k + ct$ uniform over $\mathbb{Z}_q \setminus \{ct\}$; rejecting $\widehat{R} = 1$ in the simulation excludes exactly $s = ct$. Thus, the joint distributions agree at a fresh hash input.

Step 2: Sample $\sigma_x \leftarrow \mathbb{G}$ and a uniform challenge h, and compute $\widehat{Z}_x = e(\sigma_x, \widetilde{Y}) \left(\prod_i \widehat{T}_{vk,i} \right)^{-h}$. Resample if the commitment is the identity or the corresponding H_4 input already has an answer. Otherwise, record h at $(msg, \widehat{Z}_x, (\widehat{T}_{vk,i})_i)$ and output the signature. For fixed h, this map is bijective. Rejecting the identity makes $\widehat{Z}_x$ uniform over $\mathbb{G}_t^*$, as in the real sampling of nonzero x, with the unique corresponding response $\sigma_x = (\prod_i T_{sk,i})^h X^x$. This also covers an identity aggregate temporary public key.

For q_s adaptive signing queries, the probability that real random commitments hit previously answered inputs, together with bounded-resampling error, gives

$$\mathrm{SD}(\mathsf{View}_{\mathrm{real}}, \mathsf{View}_{\mathrm{sim}}) \leq O\left(\frac{q_s(Q_3 + Q_4 + q_s)}{q - 1} \right) + \mathrm{negl}(\lambda).$$

All previous answers remain consistent, the simulator uses only public data, and retries have a polynomial bound. Hence TA-TWS achieves statistical witness hiding. □

Identity-Hiding Experiment. Witness hiding ensures that TA-TWS signatures disclose no information beyond the public inputs; it does not by itself establish that public tags hide user identities. We therefore analyze the identity hiding of fresh tags and their signatures through a separate experiment.

The challenger generates two registered users U_0, U_1 and publishes their public keys. The adversary selects the issuer set, service information, validity period, list length, and request message common to both challenge cases. The challenger uniformly samples $b \in \{0, 1\}$, generates a fresh rotation batch and tag list for U_b, and returns the externally visible tags and their TA-TWS signature. The adversary outputs a guess b'. The experiment and query interfaces are specified below.

$\mathbf{Exp}^{\mathrm{ID}}_{\Pi,\mathcal{A}}(1^\lambda)$

1: $pp \leftarrow \mathsf{Setup}(1^\lambda)$
2: $(\mathcal{SK}, \mathcal{PK}) \leftarrow \mathsf{KGen}(pp)$
3: Run the pre-challenge phase of $\mathcal{A}^{\mathcal{O}}(pp, \mathcal{PK})$
4: Receive the common challenge parameters and msg^*
5: $b \xleftarrow{\$} \{0, 1\}$
6: $st^* \leftarrow \mathsf{FreshState}$
7: Generate $\mathcal{T}^*$ and complete witnesses for U_b at st^*
8: $\Sigma^* \leftarrow \mathsf{Sign}($ $pp, \{w_i^*\}, \mathcal{T}^*, msg^*)$
9: $b' \leftarrow \mathcal{A}^{\mathcal{O}}($ $\mathcal{T}^*, msg^*, \Sigma^*)$
10: **return** $[b' = b]$

$\mathcal{O}^{\mathrm{Wit}}(T)$

1: If T belongs to the challenge batch, **return** $\perp$
2: $w \leftarrow \mathsf{WitGen}(T)$
3: **return** w

$\mathcal{O}^{\mathrm{Sign}}(\mathcal{T}, msg)$

1: If $\mathcal{T}$ is invalid or contains a challenge-batch tag, **return** $\perp$
2: Obtain complete witnesses for all tags internally
3: If any $w_i = \perp$, **return** $\perp$
4: $\Sigma \leftarrow \mathsf{Sign}($ $pp, \{w_i\}, \mathcal{T}, msg)$
5: **return** Σ

Here $\mathcal{O}$ comprises the witness and signing interfaces above and the random-oracle interface in Section IV-C. These interfaces retain their corresponding query records and update rules. The omitted honest tag- and witness-generation steps follow the construction; $\mathsf{FreshState}$ selects a new derivation state for the challenge issuance round. Before and after the challenge, the adversary may adaptively obtain complete witnesses and TA-TWS signatures for other tags and observe historical sessions and multiple verifiers. Repeated queries preserve existing records. The challenge batch uses a fresh derivation state. Neither its complete witnesses nor its associated issuance-to-identity mapping are returned to the adversary. The long-term secrets of both candidate users, the challenge tags' generation randomness, and the RA's tracing secret remain protected. The identity-hiding advantage is

$$\mathrm{Adv}^{\mathrm{ID}}_{\mathcal{A}} = \left| \Pr[b' = b] - \frac{1}{2} \right|.$$

Theorem 3 (Conditional Identity Hiding). *Under the preceding experiment conditions, if the CBDH assumption holds and f satisfies the stated PRF security, then fresh tags and their TA-TWS signatures achieve computational identity hiding.*

Proof. First, replace base-point derivation with a consistent random function, changing the adversary's success probability by at most Δ_f. The challenge tags meet Lemma 1's protection conditions. Before the first query on a genuine hidden input, their $T_{sk,i}$ can equivalently be generated as independent uniform group elements, preserving $\widehat{T}_{vk,i} = e(T_{sk,i}, \widetilde{Y})$.

By the CBDH reduction $\mathcal{B}_1$ in Lemma 1, the probability μ_1 of a genuine-hidden-input hit is negligible. Challenge-tag prefixes use independent random exponents t_i and have distributions independent of b. For any nonzero user secret usk_{U_b}, the map $v_i \mapsto usk_{U_b} v_i$ is bijective over $\mathbb{Z}_q$. Uniform hash values v_i thus produce uniform temporary secrets and pairing-based public keys whose joint distribution is independent of the candidate user. Other public fields have the same distribution for both candidates by the experiment conditions.

Next, replace the challenge TA-TWS signature with one generated by the public-input simulator $\mathcal{S}$ from Theorem 2. $\mathcal{S}$ requires no user secret, shared base, or credential. In the resulting game, the challenge cryptographic view is independent of b. Together with identical public-metadata distributions in the two cases, the adversary therefore guesses b with probability

$1/2$. Accounting for base-point replacement, genuine-hidden-input hits, signature simulation, and accidental collisions leaves only negligible identity-hiding advantage. □

Corollary. *Under the preceding protection conditions, an adversary cannot efficiently determine whether two sessions using fresh tags originate from the same user.*

Consider two experiments in which both sessions originate from U_0 in the same-user experiment, while they originate from U_0 and U_1, respectively, in the different-user experiment. Both sessions use fresh rotation batches, derivation states, and tags. Public service information, validity periods, request messages, and session metadata have identical distributions in both experiments, while candidate-user long-term secrets, challenge-tag generation randomness remain protected.

Proof. Using the hidden-input bound established by the reduction $\mathcal{B}_1$ in Lemma 1, replace each protected fresh tag's temporary secret with an independent uniform group element while preserving its pairing-based public key. Then replace the TA-TWS signatures with signatures generated by the public-input simulator $\mathcal{S}$ from Theorem 2. Whether both sessions originate from one user or from different users, the fresh random tag prefixes, temporary public keys, and simulated signatures have the same joint distribution. The experiments are therefore computationally indistinguishable, and the adversary cannot efficiently determine from these cryptographic data whether the sessions share a user. A sequence of hybrids and a union bound extend the result to polynomially many fresh rotation sessions, with negligible accumulated loss. □

D. Security Analysis in Dynamic UAV Environments

Under the preceding threat model and security experiment, we analyze cross-service interactions, asynchronous credential distribution, and tag rotation, and bound the error probability of multi-user batch verification. The conclusions retain the stated key-generation, secret-protection, and query rules.

Asynchronous Credential Distribution Across Issuers. During cross-region UAV flights and successive service accesses, an adversary may eavesdrop on communications between UAVs and service providers or base stations, collect historical authentication sessions, and gradually obtain service tags, partial credentials, complete witnesses, and TA-TWS signatures through credential or witness leakage permitted by the threat model. We use the oracle interfaces in Fig. 1 of Section IV-C to uniformly model these information acquisition capabilities. For queries satisfying the specified rules, the corresponding oracles return the defined outputs, abstracting away the specific means and difficulty of obtaining such information.

These interfaces grant the adversary capabilities beyond merely observing public communications. For example, the witness oracle allows direct access to complete witnesses associated with other service tags of the same UAV. The signing oracle enables the adversary to obtain valid signatures for selected tag lists and request messages without specifying the underlying compromise or interaction process. This models the adversary's ability to accumulate information across different times, services, and base-station sessions and adapt subsequent attacks based on previously obtained information.

The experiment does not reveal entities' long-term private keys but permits the disclosure of complete witnesses for selected tags. Its unforgeability guarantee requires that at least one tag in the final tag list has not had its complete witness returned by the witness oracle and that the final tag-list and message pair has never been returned by the signing oracle, preventing direct replay of previously obtained signatures from being counted as successful forgeries. If all required complete witnesses have been disclosed, the adversary can generate a signature using the legitimate signing algorithm, which falls outside the scope of this guarantee. This is consistent with the principle that standard signature security relies on protecting signing secrets. Under the stated model and assumptions, Theorem 1 guarantees that the probability of a successful forgery satisfying these conditions is negligible.

Security of Tag Rotation and Witness Recovery. Under Theorem 3's experiment conditions, with fresh derivation states for all relevant tags and protected long-term secrets, an external observer cannot efficiently identify a UAV or link rotation sessions solely from fresh tags and TA-TWS signatures. Local witness recovery adds no public cryptographic information beyond the protocol's prescribed outputs.

Theorem 3 establishes conditional identity hiding, and Corollary 1 makes fresh rotation sessions from the same user or different users computationally indistinguishable. Fresh tags and temporary public keys therefore do not reveal whether the sessions share a UAV. Reusing old tags or temporary public keys still permits session linking, although such linking does not itself reveal the real identity. Avoiding this direct linkage requires using different service tags during privacy rotation.

Witness recovery takes place locally on the UAV. Using its long-term secret, the user computes

$$v_i = H_1\Big(X_{t_i}, \widetilde{Y}_{t_i}, e(PK_R, \widetilde{Y}_{t_i}^{usk_U})\Big), T_{sk,i} = X^{usk_U v_i}$$

and removes credential blinding using $H_{\text{blind}}(\widetilde{A}_i^{usk_U})$. Neither long-term secret keys, hidden hash inputs, hash scalars, nor temporary secrets are sent to the base station; no additional recovery interaction is introduced. If recovery executes in a protected local environment, exposes only the prescribed tags and signatures, clears secret intermediate values no longer needed, and avoids externally visible logs, it does not additionally disclose protocol secrets or identity associations. This conclusion does not cover extra side-channel observations such as execution time, power consumption, or continuous location.

Security of Large-Scale Batch Verification. With requests and signatures fixed before the verifier privately samples batching coefficients, an adversary's probability of making invalid verification equations cancel in a batch is at most 2^{-128}. Formally, we state the result as follows:

Theorem 4. *After receiving and fixing N authentication requests and their TA-TWS signatures that satisfy format and group-membership requirements, the verifier independently and privately samples $\delta_j \leftarrow \{0, \ldots, 2^{128} - 1\}$. For prime group order $q > 2^{128}$, if at least one request fails either its authorization or message-binding equation, the probability*

that the batch passes both checks is at most 2^{-128}.

Proof. Select the family of verification equations containing an error, and express each equation's left-to-right ratio as $\widehat{Z}^{d_j}$. Valid equations have $d_j = 0$; invalid equations have $d_j \neq 0$. These exponents are used only in the analysis; the verifier need not compute discrete logarithms. Acceptance of this batch check is equivalent to $\sum_{j=1}^{N} \delta_j d_j = 0 \pmod{q}$. At least one $d_{j_0} \neq 0$. Fixing all other coefficients, prime q gives at most one solution for δ_{j_0} modulo q. Since $q > 2^{128}$, at most one value in its sampling range satisfies the equation. Thus, this erroneous check passes with probability at most 2^{-128}.

Batch acceptance requires both the authorization and message-binding checks to pass, so its event is contained in acceptance of the erroneous check. The bound remains 2^{-128} even when both checks use the same coefficients. □

Security of Eavesdropping, Tampering, and Replay. Under the threat model, observing historical authentication sessions does not enable a valid TA-TWS forgery for a fresh valid tag-list and request-message pair. An adversary may intercept tags and signatures from different UAVs, combine historical information, or modify requests and forward them to other base stations. Signing interfaces capture signature collection, while witness interfaces permit disclosure of other services' complete witnesses. By Theorem 1, these observations do not enable forgery for a fresh pair while the target service's complete witness remains protected.

For replay attack, a timestamp can be bound to the request message $msg = \mathrm{enc}(\mathrm{timestamp}, \mathrm{payload})$, where payload includes the target verifier, service request, and unique request identifier. Tampering with these fields changes the signed message, and generating a valid signature for the changed message is subject to Theorem 1. The verifier checks the timestamp against the validity window and records accepted request identifiers to reject repetitions within that window. When these application-layer checks are correctly enforced, message binding, validity windows, and deduplication jointly prevent request tampering and repeated execution without changing the TA-TWS cryptographic construction.

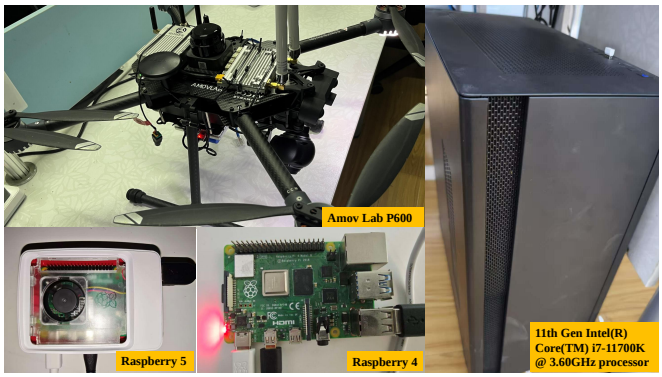
VI. Experimental Evaluation

We performed a comparative analysis against representative related schemes, namely NTAT [9], DTACB [26], and IhMA [5]. Furthermore, we investigated the protocol's authentication latency across varying network conditions.

A. Experimental Setup

We emulated users under diverse scenarios using multiple devices, including the AmovLab Prometheus 600 UAV [47], Raspberry Pi 4, and Raspberry Pi 5, as shown below.

Cryptographic Implementation: The cryptographic operations and networking logic were implemented in C++. Core cryptographic primitives were built using the MIRACL Core library [48] over the BLS12-381 elliptic curve, providing 128-bit security for bilinear pairings. We integrated the GNU Multiple Precision Arithmetic Library (GMP) with custom bridges between GMP integers and MIRACL's BIG integers.



Hashing was performed using SHA-256. Network communication was implemented via the WebSocket++ library [49]. For reproducibility, we developed a self-contained simulation framework based on Linux network namespaces and virtual Ethernet pairs, enabling single-machine topology emulation.

Hardware Environment and Network Topology: We instantiated a distributed cloud-edge testbed to map the four primary system entities: the Regulator, the Issuer, the User, and the Verifier. The regulator and the issuer were hosted remotely on an Alibaba Cloud ECS server of type ecs.e-c1m1.large, equipped with dual vCPUs and running Ubuntu 22.04. The User component was deployed across three representative edge devices—an AmovLab P600 UAV, a Raspberry Pi 4, and a Raspberry Pi 5—to evaluate performance under heterogeneous hardware conditions. The verifier was implemented within a bridged Ubuntu virtual machine on a local desktop with an Intel Core i7-11700K processor, 8 GB RAM, and Ubuntu 24.04.1. For communication, the user and the verifier were deployed within the same wireless local area network (WLAN). To systematically evaluate the protocol's performance under degraded network conditions, we introduced controlled bandwidth limitations and network delays using the Linux `tc`.

B. Computation Cost Analysis

We measured proof generation costs on multiple platforms, while verification latency was evaluated on a Host PC, as verification is typically performed by edge nodes with superior computational resources. The results are shown in Fig. 5.

Fig. 5 illustrates the overhead of generating presentation proofs for different schemes across three edge devices with varying computational capabilities when terminals concurrently request varying numbers of services. It can be observed that the variation trends remain largely consistent across all platforms. Benefiting from the proposed aggregated proof mechanism, the computational overhead of TA-TWS exhibits a more moderate growth as the number of services increases, consistently outperforming NTAT and IhMA in concurrent multi-service scenarios.

From a purely computational perspective, DTACB achieves the lowest local overhead due to its constant-time presentation algorithm. However, this advantage comes at the cost of substantial communication overhead in practice. Specifically, it relies on an independent judge for accumulator updates,

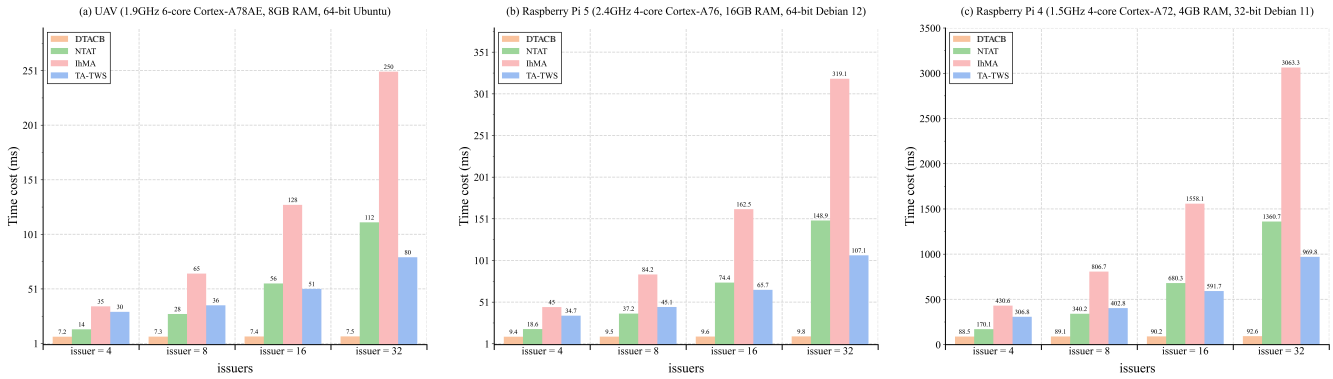


Fig. 5. Time overhead of identity proof generation overhead against the number of issuers on various computational nodes.

where each new user triggers global witness updates, requiring an additional communication round to obtain the latest witness before proof generation. As shown in Section VI-D, in highly dynamic distributed networks, even a modest inter-node latency of 50 ms can introduce over 100 ms of delay per interaction round. Consequently, DTACB lacks a competitive advantage in end-to-end latency. In contrast, TA-TWS achieves a better balance among issuance flexibility, functional completeness, and overall system latency. Notably, beyond computational efficiency, it is the only scheme listed in Table I that supports both accountability and non-interactive authorization, thereby providing stronger regulatory capability and greater deployment flexibility while maintaining high local efficiency with negligible overhead.

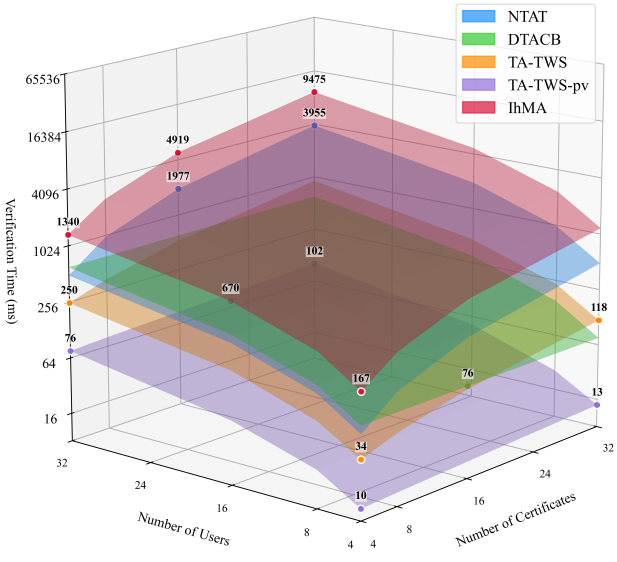


Fig. 6. Time overhead across varying numbers of users and certificates.

Compared with the credential generation time on the UAV side, the number of credentials that can be verified per unit time at the verifier side is more critical, as it directly determines the service capacity of a base station. Fig. 6 presents the computational overhead on the verification side. Since the proposed scheme supports not only the aggregated presentation of multiple credentials for a single user but also the batch verification of aggregated credentials from multiple users, it consistently achieves lower latency. Furthermore, the figure illustrates an optimized partial verification mode, denoted as TA-TWS-pv. This mode is designed for scenarios where a user continuously accesses services at the same base station. Because the structural legitimacy of the tags has already been validated during the initial interaction, the base station can safely omit these repetitive checks for subsequent requests, solely verifying whether the user possesses the private key corresponding to the tag. This strategy drastically reduces the computational burden. Specifically, when each user requests 32 services and 32 users simultaneously submit requests, the partial verification process requires only about 102 ms, which is the lowest among all compared schemes and significantly outperforms NTAT and DTACB.

Cache storage and scalability. We assume that each UAV accesses 32 distinct services on average and that each verified service tag is cached at the edge verifier. Each hash-bucket slot stores one compressed $\mathbb{G}_t$ element $\hat{T}_{vk}$ (288 B), an 8 B cache index, and an 8 B expiration timestamp, totaling 304 B. The 8 B index is used only for cache accounting.

A 100 MB cache provides approximately 328,947 hash slots, corresponding to about 246,710 valid $\hat{T}_{vk}$ entries at a load factor of 0.75. If each service tag has a cache lifetime of τ hours, the steady-state capacity is $N(\tau) = 246{,}710/(32\tau)$ new UAVs per hour. Thus, with cache lifetimes of 1 and 4 hours, one edge verifier can accommodate approximately 7,709 and 1,927 new UAVs per hour, respectively. Cache expiration affects only the fast-verification result, not credential validity. If the credential remains valid, the next request is fully verified once and can then be cached again. Therefore, the cache lifetime can be adjusted according to available storage, UAV arrival rate, and the desired cache hit rate.

Under the same assumptions, increasing the cache allocation from 100 MB to 1 GB raises the 4-hour capacity to approximately 19,270 new UAVs per hour. These results indicate that caching $\hat{T}_{vk}$ can support high UAV arrival rates with modest storage overhead. The estimates exclude implementation-specific hash-table metadata and allocator overhead.

C. Signature Size Analysis

In this section, we analyze the signature overhead of the proposed TA-TWS scheme in comparison with NTAT, IhMA, and DTACB. We focus on the size of both a single-service credential (partial signature) and the aggregated credential over n services (aggregate signature), as summarized in Table III.

TABLE III
SIGNATURE SIZE COMPARISON OF FOUR SCHEMES

Scheme \ Phase	Single-service Credential	n-service Credential
DTACB	$4G_1 + 4G_2 + G_t + 5Z_p$	$4G_1 + 4G_2 + G_t + 5Z_p$
NTAT	$G_1 + 5Z_q$	$n \times (G_1 + 5Z_q)$
IhMA	$5G_1 + 2G_2 + 3Z_q$	$(n+4)G_1 + 2nG_2 + 3Z_q$
TA-TWS	$3G_1 + G_2 + 2G_t + Z_q$	$3G_1 + G_2 + 2G_t + Z_q$

We estimate concrete signature sizes based on the BLS12-381 curve with standard point compression. A $\mathbb{G}_1$ element occupies 49 B, while a $\mathbb{G}_2$ element requires 97 B. Elements in $\mathbb{G}_t$ can be compressed into $\mathbb{F}_{p^6}$ due to their cyclotomic subgroup structure, resulting in a size of 288 B. Scalars in $\mathbb{Z}_q$ or $\mathbb{Z}_p$ occupy 48 B each.

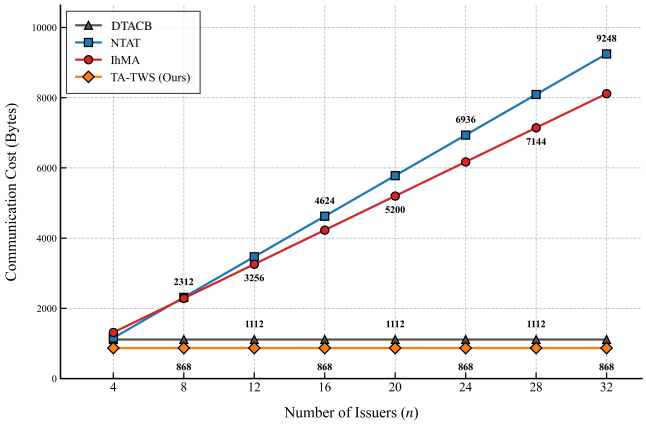


Fig. 7. Comparison of signature sizes under varying numbers of issuers (n).

As shown in Table III and Fig. 7, both DTACB and TA-TWS achieve constant-size aggregate signatures, yielding an optimal $O(1)$ signature overhead. In particular, TA-TWS maintains a fixed size of 868 B regardless of the number of aggregated credentials. In contrast, NTAT and IhMA exhibit linear growth ($O(n)$), with sizes of $289n$ B and $243n + 340$ B, respectively.

Although the partial signature of TA-TWS is larger than that of NTAT due to the inclusion of $\mathbb{G}_t$ elements, this overhead is amortized by its aggregation capability. As illustrated in Fig. 7, once $n > 3$, the total signature size of TA-TWS becomes smaller than that of both NTAT and IhMA, and remains consistently below that of DTACB (1112 B). This constant-size property is particularly advantageous in resource-constrained systems, as it significantly reduces bandwidth consumption and enhances scalability under high-concurrency authentication.

D. Impact of Network Conditions on Authentication

Following the computational micro-benchmarks, we conducted end-to-end authentication latency tests within the constructed network topology. The total latency includes proof generation, network transmission, and remote verification. To evaluate performance under adverse conditions, we introduced controlled network.

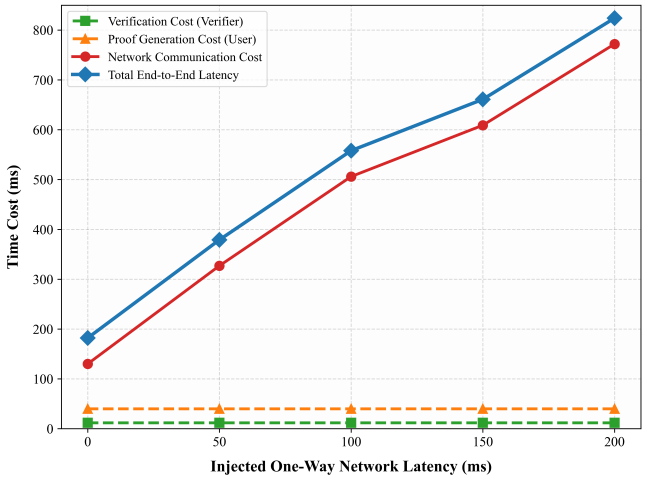


Fig. 8. Breakdown of authentication latency under varying network latencies.

As depicted in Fig. 8, the cryptographic operations are highly stable: the proof generation time on the UAV remains around 40 ms, and the verification cost at the verifier stays at approximately 12 ms, both largely unaffected by network conditions. In contrast, the end-to-end latency is dominated by network communication.

Experimental results show a clear positive linear correlation between the total communication overhead and the injected one-way network latency. This overhead mainly stems from the interaction delays inherent in underlying network protocols, such as TCP and HTTP, rather than cryptographic computation. These findings highlight that, in practice, communication rounds have a more direct impact on user experience than computational cost, and that the primary bottleneck in large-scale distributed authentication systems lies in network interactions. Consequently, reducing interaction rounds, for example through long-lived keep-alive connections, can significantly improve overall performance.

Finally, to simulate extreme edge communication bottlenecks, we imposed strict bandwidth constraints ranging from 128 kbit to 1024 kbit, as shown in Fig. 9. Under severely restricted bandwidth (128 kbit) and a high number of issuers ($I = 32$), the total authentication time spikes significantly, peaking at 1957 ms due to network congestion and packet serialization delays. However, the system demonstrates strong resilience: once the bandwidth threshold reaches 256 kbit (comparable to basic edge communication standards), the latency curve flattens dramatically. At 256 kbit and beyond, the authentication times converge rapidly to optimal levels across all issuer counts, proving that TA-TWS requires only minimal throughput to guarantee seamless and swift authentication in real-world aerial networks.

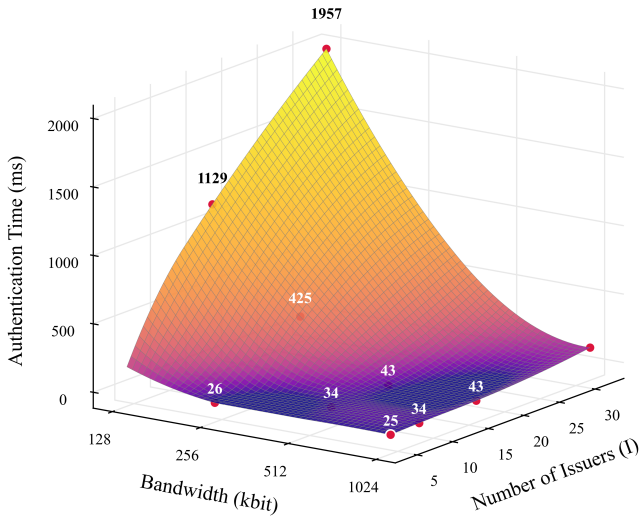


Fig. 9. Total authentication time under varying bandwidth constraints and number of issuers (I).

To control variations in packet loss across network paths and over time, we constructed the experimental topology using network namespaces and injected preset random packet loss with Linux `tc netem`. The number of issuers was set to $N \in \{4, 8, 16, 32\}$, and the packet loss rates were set to 0%, 1%, 2%, 4%, and 6%. Each parameter combination was repeated 30 times, yielding 600 valid runs. We define authentication time as the elapsed time from the start of proof generation by the user to the receipt of the verification result.

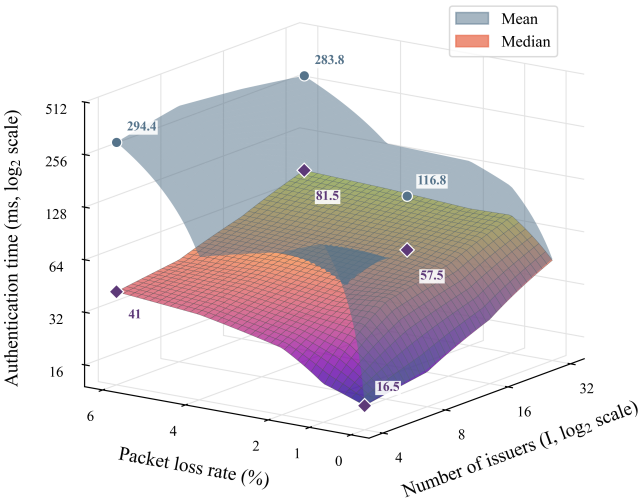


Fig. 10. Mean and median authentication time under different packet loss rates and numbers of issuers.

Fig. 10 shows the mean and median authentication times for each parameter combination. The blue-gray surface represents the mean, and the warm-brown surface represents the median. For $N = 32$, as the packet loss rate increases from 0% to 6%, the mean authentication time increases from 45.8 ms to 283.8 ms, whereas the median increases from 45.0 ms to 81.5 ms. The substantial gap between the two at higher packet loss rates indicates that a small number of slow runs markedly increase the mean. Thus, the mean alone does not adequately describe the time required for a typical authentication. To examine the variation across individual runs, Fig. 11 presents their empirical cumulative distribution functions.

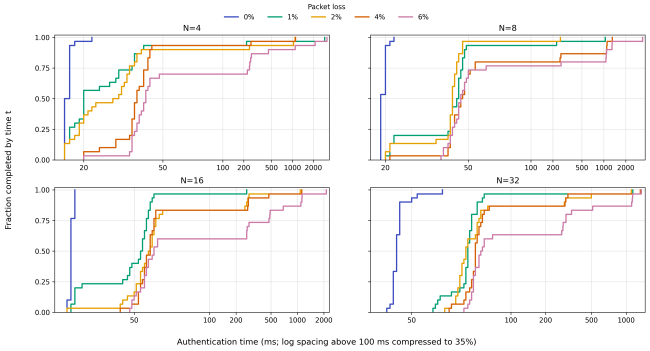


Fig. 11. Empirical cumulative distributions of authentication time under different packet loss rates and numbers of issuers.

In Fig. 11, the horizontal axis shows authentication time, and the vertical axis shows the fraction of runs completed within that time. At a given time, a higher curve therefore indicates that a larger fraction of runs has completed. Without packet loss, authentication times are concentrated within a narrow range. At higher packet loss rates, the difference between faster and slower runs becomes more pronounced. For example, when $N = 32$ and the packet loss rate is 6%, 19 of the 30 runs finish within 200 ms and the median is 81.5 ms, while the 90th percentile reaches 1116.3 ms. Higher packet loss therefore does not add the same delay to every run; instead, some runs become substantially slower. This variation also explains the gap between the mean and median.

E. Energy Consumption Analysis

We further evaluate the energy overhead of proof generation on the UAV platform. The experiments are conducted on the same Amov P600 UAV equipped with an NVIDIA Jetson Xavier NX. Power consumption is sampled using `tegrastats`, where the `VDD_IN` value is used as the platform-level power measurement. Since a single proof generation completes within a relatively short period, directly integrating the instantaneous power of one execution would be sensitive to the sampling interval. Therefore, we repeatedly execute the target algorithm to create a stable computation period and measure its average power consumption.

Specifically, we first record the average idle power P_{idle} when no benchmark process is running. We then continuously execute the proof generation procedure and record the average power P_{active} over the stable high-load interval. Transient samples corresponding to process startup and termination are excluded. From the measurements, the average dynamic power consumed by cryptographic computation is approximately

$$\Delta P = P_{\text{active}} - P_{\text{idle}} \approx 0.584 \text{ W}.$$

Accordingly, for a proof-generation latency of t milliseconds, its energy consumption can be estimated as $E = \Delta P \cdot t$,

where E is directly expressed in mJ since $\mathrm{W} \cdot \mathrm{ms} = \mathrm{mJ}$. For consistency, the same measured dynamic-power coefficient is applied to all schemes, while their respective execution times are obtained from the latency experiments.

TABLE IV
ENERGY CONSUMPTION (MJ) UNDER DIFFERENT NUMBERS OF ISSUERS.

Scheme	4	8	16	32
DTACB	4.2	4.2	4.3	4.4
NTAT	8.2	16.4	32.7	65.4
IhMA	20.4	38.0	74.8	146.0
TA-TWS	17.5	21.0	29.8	46.7

Table IV shows that the energy consumption of TA-TWS increases from 17.5 mJ with four issuers to 46.7 mJ with 32 issuers, remaining below 0.05 J even in the largest setting. Comparisons with other schemes yield conclusions consistent with the computational cost analysis and are therefore not discussed further. Given a typical UAV flight power of 600–800 W, the maximum energy consumption of TA-TWS (0.04672 J) is equivalent to only 0.058–0.078 ms of normal flight operation. Thus, its energy overhead accounts for a negligible fraction of the overall UAV energy budget.

VII. CONCLUSION

In this paper, we proposed a traceable and aggregatable tag-based witness signature (TA-TWS) scheme, which transforms interactive credential issuance into an asynchronous witness recovery process. This design removes the reliance on synchronous connectivity and long-term public keys during presentation, achieving a strong balance between user anonymity and regulatory accountability through an embedded cryptographic trapdoor. The TA-TWS-based framework enables anonymous multi-service authentication in dynamic UAV networks through cross-issuer credential aggregation and verifier-side batch verification. A regulatory trapdoor provides conditional traceability without exposing users' long-term public keys to verifiers during authentication.

We proved unforgeability and witness hiding under the stated model and assumptions. A C++ implementation using MIRACL Core and experiments on UAV and edge platforms demonstrate the scheme's computational and communication efficiency in concurrent multi-service access.

The current construction relies on a trusted regulator holding the tracing key. Future work will explore threshold tracing, enabling multiple authorities to jointly authorize identity opening and improve resilience while preserving aggregation and batch verification. This requires distributed key generation, threshold-compatible pseudorandom-base derivation, and tracing protocols, together with collusion and availability analyses. We will also investigate fine-grained credential revocation and multi-issuer key management under asynchronous updates.

REFERENCES

[1] Q. Zhang, B. Ren, H. Zhong, F. Wang, M. Zeng, and J. Cui, "Distributed multi-attribute anonymous certificate management scheme with fine-grained revocation for unmanned aerial vehicles," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–14, 2026.

[2] M. Zeng, H. Zhong, Q. Zhang, F. Wang, J. Li, and J. Cui, "Runtime threshold signature-based unmanned aerial vehicle swarm authentication with autonomous splitting support," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–17, 2026.

[3] L. Rui, X. Zhang, J. Yang, and W. Xu, "Efficient handover authentication protocol for cloud-edge collaborative UAV networks," *Journal on Communications*, vol. 46, no. 6, pp. 60–72, 2025.

[4] H. Li, H. Li, C. Ma, J. Guan, J. Zeng, H. Feng, and Z. Zhao, "5g-rnaka: A random number-based authentication and key agreement protocol for 5g systems," in *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, CCS '25, (New York, NY, USA), p. 1634–1648, Association for Computing Machinery, 2025.

[5] O. Mir, B. Bauer, S. Griffy, A. Lysyanskaya, and D. Slamanig, "Aggregate signatures with versatile randomization and issuer-hiding multi-authority anonymous credentials," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, CCS '23, (New York, NY, USA), p. 30–44, Association for Computing Machinery, 2023.

[6] M. Jia, J. Chen, Y. Wang, K. He, M. Shi, and R. Du, "Multi-authority anonymous credentials with efficient and decentralized supervision," *IEEE Transactions on Information Forensics and Security*, vol. 20, pp. 8154–8166, 2025.

[7] A. Flamini, E. Lee, and A. Lysyanskaya, "Multi-holder anonymous credentials from bbs signatures," in *Advances in Cryptology – CRYPTO 2025* (Y. Tauman Kalai and S. F. Kamara, eds.), (Cham), pp. 325–357, Springer Nature Switzerland, 2025.

[8] B. Xie, R. Song, Z. Li, X. Deng, and B. Xiao, "Flexible and privacy-preserving access control framework for decentralized identity systems," *IEEE Transactions on Information Forensics and Security*, pp. 1–1, 2026.

[9] F. B. Durak, L. Marco, A. Talayhan, and S. Vaudenay, "Non-transferable anonymous tokens by secret binding," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, CCS '24, (New York, NY, USA), p. 2460–2474, Association for Computing Machinery, 2024.

[10] X. Xu, C. Fu, X. Du, and B. Luo, "Discovering and exploiting iot device hidden attributes: A new vulnerability in smart homes," in *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, CCS '25, (New York, NY, USA), p. 1649–1663, Association for Computing Machinery, 2025.

[11] S. Shamshad, S. Belguith, and A. Oracevic, "Drones don't trust blindly: Quantum-secure AKE protocol for IoD-enabled FANETs," *IEEE Transactions on Intelligent Transportation Systems*, vol. 27, no. 8, pp. 10292–10305, 2026.

[12] A. Kumar, M. Cunche, and H. H. Arcolezi, "How tough is location anonymization? re-identifying 100k real-user trajectories in japan," in *Proceedings of the 21st ACM ASIA Conference on Computer and Communications Security (ASIACCS)*, 2026.

[13] O. Mir, D. Slamanig, B. Bauer, and R. Mayrhofer, "Practical delegatable anonymous credentials from equivalence class signatures," *Proceedings on Privacy Enhancing Technologies*, 2023.

[14] M. Shih, M. Rosenberg, H. Kailad, and I. Miers, "Zk-promises: Anonymous moderation, reputation, and blocking from anonymous credentials with callbacks," in *34th USENIX Security Symposium (USENIX Security 25)*, pp. 4995–5014, 2025.

[15] R. Shi, H. Feng, Y. Yang, Y. Li, X. Li, and R. H. Deng, "Hecate: Threshold anonymous credentials with private verifiers and issuer-hiding," *IEEE Transactions on Dependable and Secure Computing*, vol. 23, no. 1, pp. 1160–1172, 2026.

[16] M. Xie, P. Ju, Y. Zhao, Z. L. Jiang, M. H. Au, J. Fang, Y. Yu, and X. Wang, "Fdaac-cr: Practical delegatable attribute-based anonymous credentials with fine-grained delegation management and chainable revocation," *IEEE Transactions on Information Forensics and Security*, vol. 20, pp. 7572–7587, 2025.

[17] M. Sun, J. Lai, X. Mo, C. Wu, P. Li, C.-K. Chu, and R. H. Deng, "Delegatable multi-authority attribute-based anonymous credentials," *IEEE Transactions on Cloud Computing*, vol. 13, no. 2, pp. 655–666, 2025.

[18] Y. Liu, D. He, M. Luo, H. Wang, and Q. Liu, "Atrc: An anonymous traceable and revocable credential system using blockchain for vanets," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 2, pp. 2482–2494, 2024.

[19] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," in *Advances in Cryptology – CRYPTO 2004* (M. Franklin, ed.), (Berlin, Heidelberg), pp. 41–55, Springer Berlin Heidelberg, 2004.

[20] L. Zhang, J. Li, and Y. Yang, "Message linkable group signature with information binding and efficient revocation for privacy- preserving

announcement in vanets," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 6, pp. 5667–5680, 2024.
[21] M. Zeng, J. Cui, Q. Zhang, H. Zhong, and D. He, "Efficient revocable cross-domain anonymous authentication scheme for iiot," *IEEE Transactions on Information Forensics and Security*, vol. 20, pp. 996–1010, 2025.
[22] B. Libert, T. Peters, and M. Yung, "Short group signatures via structure-preserving signatures: standard model security from simple assumptions," in *Annual Cryptology Conference*, pp. 296–316, Springer, 2015.
[23] F. Baldimtsi, K. K. Chalkias, Y. Ji, J. Lindstrøm, D. Maram, B. Riva, A. Roy, M. Sedaghat, and J. Wang, "zklogin: Privacy-preserving blockchain authentication with existing credentials," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, CCS '24, (New York, NY, USA), p. 3182–3196, Association for Computing Machinery, 2024.
[24] Z. Liu, G. Yang, D. S. Wong, K. Nguyen, H. Wang, X. Ke, and Y. Liu, "Secure deterministic wallet and stealth address: Key-insulated and privacy-preserving signature scheme with publicly derived public key," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 5, pp. 2934–2951, 2022.
[25] S. Griffy, A. Lysyanskaya, O. Mir, O. Perez Kempner, and D. Slamanig, "Delegatable anonymous credentials from mercurial signatures with stronger privacy," in *Advances in Cryptology – ASIACRYPT 2024* (K.-M. Chung and Y. Sasaki, eds.), (Singapore), pp. 296–325, Springer Nature Singapore, 2025.
[26] C. Li, J. Ning, S. Xu, C. Lin, J. Li, and J. Shen, "Dtacb: Dynamic threshold anonymous credentials with batch-showing," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 7744–7758, 2024.
[27] R. Chairattana-Apirom, F. Harding, A. Lysyanskaya, and S. Tessaro, "Server-aided anonymous credentials," in *Advances in Cryptology – CRYPTO 2025* (Y. Tauman Kalai and S. F. Kamara, eds.), (Cham), pp. 291–324, Springer Nature Switzerland, 2025.
[28] A. Connolly, P. Lafourcade, and O. Perez Kempner, "Improved constructions of anonymous credentials from structure-preserving signatures on equivalence classes," in *Public-Key Cryptography – PKC 2022* (G. Hanaoka, J. Shikata, and Y. Watanabe, eds.), (Cham), pp. 409–438, Springer International Publishing, 2022.
[29] G. Fuchsbauer, C. Hanser, and D. Slamanig, "Structure-preserving signatures on equivalence classes and constant-size anonymous credentials," *Journal of Cryptology*, vol. 32, no. 2, pp. 498–546, 2019.
[30] C. Hébant and D. Pointcheval, "Traceable constant-size multi-authority credentials," *Information and Computation*, vol. 293, p. 105060, 2023.
[31] R. Shi, H. Feng, Y. Yang, F. Yuan, Y. Li, H. H. Pang, and R. H. Deng, "Threshold attribute-based credentials with redactable signature," *IEEE Transactions on Services Computing*, vol. 16, no. 5, pp. 3751–3765, 2023.
[32] J. Doerner, Y. Kondi, E. Lee, A. Shelat, and L. Tyner, "Threshold bbs+ signatures for distributed anonymous credential issuance," in *2023 IEEE Symposium on Security and Privacy (SP)*, pp. 773–789, 2023.
[33] O. Mir, D. Slamanig, and R. Mayrhofer, "Threshold delegatable anonymous credentials with controlled and fine-grained delegation," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, pp. 2312–2326, 2024.
[34] M. Rosenberg, J. White, C. Garman, and I. Miers, "zk-creds: Flexible anonymous credentials from zksnarks and existing identity infrastructure," in *2023 IEEE Symposium on Security and Privacy (SP)*, pp. 790–808, 2023.
[35] F. Baldimtsi, K. K. Chalkias, Y. Ji, J. Lindstrøm, D. Maram, B. Riva, A. Roy, M. Sedaghat, and J. Wang, "zklogin: Privacy-preserving blockchain authentication with existing credentials," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, CCS '24, (New York, NY, USA), p. 3182–3196, Association for Computing Machinery, 2024.
[36] H. Xu, C. Hsu, L. Harn, J. Cui, Z. Zhao, and Z. Zhang, "Three-factor anonymous authentication and key agreement based on fuzzy biological extraction for industrial internet of things," *IEEE Transactions on Services Computing*, vol. 16, no. 4, pp. 3000–3013, 2023.
[37] X. Ai, A. Badshah, S. Tu, M. Waqas, and I. Ahmad, "An improved ultra-lightweight anonymous authenticated key agreement protocol for wearable devices," *IEEE Transactions on Mobile Computing*, vol. 24, no. 5, pp. 4543–4557, 2025.
[38] S. Jiang, X. Zhang, G. Sang, H. Chi, and Y. Zhou, "Ecakm: Efficient conditional anonymous authentication scheme with on-chain key management in vanets," *IEEE Transactions on Intelligent Transportation Systems*, vol. 26, no. 12, pp. 23407–23418, 2025.
[39] M. A. Al Sibahee, Z. A. Abduljabbar, V. O. Nyangaresi, J. Li, C. Luo, A. Ngueilbaye, J. Ma, Y. Huang, J. Zhang, and H. A. Neamah, "Bilinear pairing and deffie-hellman based anonymous authentication protocol for the internet of vehicles," *IEEE Transactions on Dependable and Secure Computing*, vol. 23, no. 2, pp. 2635–2647, 2026.
[40] X. Zhang, X. Chen, S. Liu, and S. Zhong, "Anonymous authentication and information sharing scheme based on blockchain and zero knowledge proof for vanets," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 12, pp. 18043–18058, 2024.
[41] Y. Yang, W. Xue, J. Sun, G. Yang, Y. Li, H. Hwa Pang, and R. H. Deng, "Pkt-sin: A secure communication protocol for space information networks with periodic k-time anonymous authentication," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 6097–6112, 2024.
[42] C. Du, H. Yu, Y. Xiao, Y. T. Hou, A. D. Keromytis, and W. Lou, "UCBlocker: Unwanted call blocking using anonymous authentication," in *32nd USENIX Security Symposium (USENIX Security 23)*, (Anaheim, CA), pp. 445–462, USENIX Association, Aug. 2023.
[43] D. Pointcheval and O. Sanders, "Short randomizable signatures," in *Topics in Cryptology - CT-RSA 2016* (K. Sako, ed.), (Cham), pp. 111–126, Springer International Publishing, 2016.
[44] D. Boneh, A. Partap, and L. Rotem, "Traceable verifiable random functions," in *Advances in Cryptology – CRYPTO 2025* (Y. Tauman Kalai and S. F. Kamara, eds.), (Cham), pp. 203–234, Springer Nature Switzerland, 2025.
[45] G. Fuchsbauer, E. Kiltz, and J. Loss, "The algebraic group model and its applications," in *Advances in Cryptology – CRYPTO 2018* (H. Shacham and A. Boldyreva, eds.), (Cham), pp. 33–62, Springer International Publishing, 2018.
[46] M. Bellare and G. Neven, "Multi-signatures in the plain public-key model and a general forking lemma," in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, CCS '06, (New York, NY, USA), p. 390–399, Association for Computing Machinery, 2006.
[47] Amovlab, "Prometheus autonomous uav opensource project." https://github.com/amov-lab/Prometheus.
[48] "miracl/core," Jan. 2025. https://github.com/miracl/core.
[49] "websoket," Jan. 2025. https://github.com/zaphoyd/websocketpp.git.