

SLAP: Secure Location-proof and Anonymous Privacy-preserving Spectrum Access

Saleh Darzi, Attila A. Yavuz

Department of Computer Science and Engineering, University of South Florida, Tampa, FL, USA
(salehdarzi@usf.edu, attilaayavuz@usf.edu)

Abstract—The rapid advancements in wireless technology have significantly increased the demand for communication resources, leading to the development of Spectrum Access Systems (SAS). However, network regulations require disclosing sensitive user information, such as location coordinates and transmission details, raising critical privacy concerns. Moreover, as a database-driven architecture reliant on user-provided data, SAS necessitates robust location verification to counter identity and location spoofing attacks and remains a primary target for denial-of-service (DoS) attacks. Addressing these security challenges while adhering to regulatory requirements is essential.

In this paper, we propose SLAP, a novel framework that ensures location privacy and anonymity during spectrum queries, usage notifications, and location-proof acquisition. Our solution includes an adaptive dual-scenario location verification mechanism with architectural flexibility and a fallback option, along with a counter-DoS approach using time-lock puzzles. We prove the security of SLAP and demonstrate its advantages over existing solutions through comprehensive performance evaluations.

Index Terms—Spectrum Access Systems, Location Privacy, Anonymous Credentials, Location Proof, Counter-DoS

I. INTRODUCTION

Spectrum Access Systems (SAS) have become the de facto technology for dynamic spectrum allocation, enabling efficient sharing among primary (PU) and secondary users (SU) while ensuring regulatory compliance and interference management. A notable example is the Citizens Broadband Radio Service (CBRS) in the U.S., operating in the 3.5 GHz band for federal and satellite services [1]. However, SAS introduces significant privacy and security challenges due to its reliance on continuous reporting of user location and transmission details to geo-location databases, raising concerns about user anonymity and privacy [2]. The location-based nature of SAS also makes it vulnerable to spoofing, location fraud, and falsified data, increasing the risk of unauthorized spectrum access [3]. Furthermore, its database-driven architecture leaves SAS and Cognitive Radio Networks (CRNs) susceptible to denial-of-service (DoS) attacks, which compromise spectrum availability and system efficiency [4]. Despite various solutions targeting privacy protection, location verification, and DoS resistance, existing approaches remain isolated and fail to address these issues comprehensively. In the following, we review relevant efforts related to our work.

A. Related Work

Location Privacy and Anonymous Spectrum Access in Database-Driven CRN: Compliance with Federal Communications Commission (FCC) regulations in centralized SAS requires the disclosure of sensitive user information, including

precise location coordinates, spectrum channel preferences, usage data, and transmission details, to query spectrum availability. This mandatory reporting raises serious privacy concerns, such as location privacy breaches, identity tracing, and the exposure of behavioral patterns. Existing location-privacy schemes often have significant limitations. Many focus solely on SUs, neglecting PUs, where their impact on spectrum information is most critical. Computationally or information-theoretically secure Private Information Retrieval (PIR) methods demand resource-intensive operations or involve extensive communication with multiple non-colluding databases, imposing high computational and communication overhead [5]–[7]. Approaches based on k -anonymity and pseudo-identifiers fail to provide provable security, offering only weak privacy guarantees unless an impractically large k value is used, which is infeasible for large-scale networks with numerous users [8], [9]. Similarly, differential privacy-based methods degrade the accuracy of spectrum availability information [10]. These shortcomings highlight the need for efficient mechanisms that ensure robust security, full anonymity, and strong location privacy against all network entities without compromising system performance and user experience.

Location Proof and Spoofing Attack Resistance in SAS: SAS, viewed as location-based services reliant on real-time user data, depend on the accuracy and integrity of this information for efficient and fair spectrum allocation. However, adversaries can exploit this reliance by impersonating legitimate entities or falsifying location and usage data to manipulate spectrum allocation, leading to spectrum interference, operational disruptions, and economic losses. Existing works addressing location proofs in SAS often fail to comprehensively mitigate broader threats, including location spoofing, distance fraud, mafia attacks, and distance hijacking [3], [11]. Many solutions rely on impractical assumptions, such as the existence of dedicated location-proof servers [12], the inherent honesty of some entities [7], or the availability of trusted infrastructure like WiFi or cellular access points in all locations. These assumptions are unrealistic, especially in rural or sparsely populated areas where such infrastructure may be absent, limiting the applicability of these methods. Also, most schemes do not safeguard location privacy and anonymity against access points or location servers, leaving a significant gap. Thus, there is a pressing need for a practical and robust location verification mechanism in SAS that ensures privacy, anonymity, and resilience against diverse attack scenarios while aligning with the operational constraints of real-world deployments.

DoS countermeasures for SAS and CRN Services: The proliferation of inexpensive devices (e.g., IoT) and the reliance of SAS on geo-location databases have significantly amplified the risk of DoS attacks [4]. These attacks overwhelm systems with malicious requests, disrupting spectrum allocation and degrading performance, particularly during spectrum usage notifications and CRN service requests. Proposed solutions include intrusion detection systems (IDSs), blockchain, cryptographic techniques like client puzzles, and game-theory-based methods [13]. While AI-based detection excels at identifying attacks, it primarily focuses on detection rather than prevention and requires extensive network-wide knowledge and access to sensitive user traffic—an impractical approach for real-time SAS countermeasures. Similarly, client-puzzle protocols face challenges such as distribution inefficiencies, parallelization vulnerabilities, and excessive overhead on servers and users, limiting their feasibility. There is an urgent need for efficient DoS countermeasures tailored to tasks like spectrum usage notifications and CRN service requests, ensuring resilience without imposing significant resource burdens.

B. Our Contributions

We developed an efficient framework that innovatively synergies advanced cryptographic techniques to address the complex privacy and security challenges posed by regulatory requirements on SAS such as DoS and spoofing attacks. The proposed scheme, “Secure Location-Proof and Anonymous Privacy-Preserving Spectrum Access (SLAP)”, is designed to meet these challenges effectively. Key desirable properties of the SLAP framework are outlined below:

- Location Privacy-Preserving and Anonymous Spectrum Access: We enable anonymous queries to geo-location databases while ensuring location privacy and compliance with FCC regulations: (i) Key operations, including spectrum queries by SUs, database population with PU information, location proof requests from WiFi access points (APs) or nearby devices, and CRN service requests, are executed using delegatable attribute-based anonymous credentials [14]. These credentials are built on structure-preserving signatures on equivalence classes with updateable commitments, ensuring unlinkability and untraceability for robust location privacy. (ii) Certified attributes in anonymous credentials containing device-specific information along with location proofs, ensure full anonymity during authentication while significantly improving the quality and reliability of SAS and CRN services. (iii) Our comparison demonstrates that SLAP achieves spectrum query with significantly lower end-to-end delay, outperforming existing schemes by a wide margin: 17× faster than [6], 63× faster than [15], 22.6× faster than [16], and 5.9× faster than [7]. Additionally, SLAP requires communication with only a single database and reduces total communication overhead by two orders of magnitude compared to PIR-based schemes that rely on multiple non-colluding databases.

- Adaptive Location Proofs and Attack Resistance for Spectrum Access: We propose an adaptive location verification algorithm with dual scenario support and architectural flexibility: (i) *AP-Based Verification:* In areas with accessible APs,

the algorithm utilizes signal strength to verify user proximity and generate location proofs using group signatures [17], ensuring privacy and anonymity. Compared to existing solutions, SLAP achieves location proof 2× faster than [12] and 4× faster than [7], while uniquely offering location privacy and full anonymity even against the AP itself. (ii) *Device-Based Verification:* In the absence of APs, the algorithm employs public key distance-bounding protocols [18] and delegatable anonymous credentials, enabling users to prove proximity to nearby devices and obtain delegated credentials with certified location proofs as attributes, all while preserving privacy and anonymity. This dual-scenario approach enhances resilience by providing a fallback mechanism, mitigating single points of failure, and ensuring robust architectural adaptability. It guarantees strong location privacy, full anonymity, and security against spoofing attacks, regardless of infrastructure availability or environmental conditions.

- Counter-DoS Mechanism for Spectrum Usage: We propose a proactive defense mechanism utilizing time-lock puzzles. These puzzles, designed to resist parallelization, are generated using encryption schemes and tailored to device-specific capabilities recorded in user credential attributes, imposing no database storage overhead unlike other counterparts [6].

II. PRELIMINARIES

This section outlines the notations, cryptographic preliminaries, and foundational components of our framework.

Notations: $|x|$ and $\{0,1\}^k$ signify the bit length of a variable and k -bit binary value, respectively. \oplus represents XOR operation. $\{x_i\}_{i=1}^\ell$ and $\overset{\$}{\leftarrow} \mathcal{S}$ denote $(x_1, x_2, \dots, x_\ell)$ and random selection from the set \mathcal{S} , respectively. Let $\mathbb{G}_1, \mathbb{G}_2$, and \mathbb{G}_T be prime-order groups with order p , and let $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ denote a bilinear map satisfying bilinearity and non-triviality. $\mathbf{m}[i]$ refers to the i -th element of the vector \mathbf{m} and $h(\cdot)$ denotes a cryptographically secure hash function. sk and pk are secret and public keys, respectively.

Delegatable Anonymous Credentials (DAC): We use an attribute-based DAC [14] for anonymous authentication, built upon structure-preserving signatures on equivalence classes of updatable commitments (SPSEQ-UC). The main algorithms are outlined below; additional details are available in [14]:

- $(pp, sk_{RI}, pk_{RI}) \leftarrow Setup(1^\lambda, 1^t, 1^\eta)$: Given the security parameter λ , an upper bound t for the set commitment scheme’s maximum cardinality, and a length parameter $\eta > 1$, it produces the system’s public parameters pp along with a signing key sk_{RI} and a public key pk_{RI} for each level $i \in [\eta]$ associated with the root issuer (RI), where pp is implicitly provided as input to all subsequent algorithms.
- $(pk, sk) \leftarrow KeyGen(pp)$: Given pp , it generates the user’s key pairs (sk, pk) , where pk is the initial pseudonym.
- $(nym, aux) \leftarrow NymGen(pk)$: Given a user’s public key pk , the algorithm generates a pseudonym nym and auxiliary information aux (randomness) for its usage.
- $CreateCred(L', A, sk_{RI}) \leftrightarrow GetCred(pk_{RI}, sk_u, A) \rightarrow (cred, (\vec{C}, \vec{O}), dk_{L'})$: An interactive algorithm between RI and a user identified by nym_u . Given pp , the RI’s public

key pk_{RI} and attribute set A , RI generates a delegatable root credential for the user via the SPSEQ-UC signature. This credential is rooted at pk_{IR} and created for a set commitment C certifying the attribute set A . The user receives the credential $cred$, the opening information O , and a delegatable key $dk_{L'}$ for level L' .

- $IssueCred(pk_{RI}, dk_{L'}, sk_u, cred_u, A_l, L'') \leftrightarrow ReceiveCred(pk_{RI}, sk_r, A_l) \rightarrow (cred_r, dk_{L''})$: This interactive algorithm involves a delegator (nym_i) and a delegatee (nym_r). The delegator uses inputs including pp , pk_{RI} , attribute set A_l , delegation key $dk_{L'}$, secret key sk_i , credential $cred_i$, and auxiliary information aux_i to generate a new credential $cred_r$. The delegatee, using their secret key sk_r and pk_{RI} , receives $cred_r$ with an extended attribute set $A' = (A, A_l)$ and a delegation level L'' satisfying $L'' \leq L'$. The new credential includes an updated delegation key $dk_{L''}$, allowing further delegation if permitted.
- $CredProve(pk_{RI}, sk_p, nym_p, aux_p, cred_p, D) \leftrightarrow CredVerify(pk_{RI}, nym_p, D) \rightarrow \{0, 1\}$: This interactive protocol allows a credential holder to anonymously prove ownership of their credential to a verifier. The prover, identified by pseudonym nym_p , uses their secret key sk_p , auxiliary information aux_p , and credential $cred_p$ to generate a proof validating $cred_p$ with respect to a disclosed attribute set D . The verifier, using the RI's public key pk_{RI} and the prover's pseudonym nym_p , verifies the proof against the disclosed attributes. If the proof is valid, the verifier outputs 1; otherwise, it outputs 0.

Distance Bounding Protocol (DBP): A DBP verifies the physical proximity of two network entities by measuring message transmission times during a rapid challenge-response exchange. We utilize a public key-based DBP [18] built on a one-pass authenticated key agreement (AKA) protocol using the nonce-Diffie Hellman scheme to establish a session key between the prover (P) and verifier (V). This is further combined with a symmetric DBP [19] operating on the session key. The DBP comprises the following algorithms:

- $ss \leftarrow AKA(sk, pk, pk')$: P and V derive the session key ss using their own key pair and the other's public key pk' .
- $\{0, 1\} \leftarrow SymDBP(ss, th)$: An interactive algorithm between P and V to verify proximity, given a distance threshold th and session key ss . (1) *Initialization Phase*: V selects $m \in \{0, 1\}^{2n}$, sends it to P . P computes $a = ss \oplus m$. (2) *Rapid Bit Exchange Phase (Time-Critical)*: V sends challenges ($c_i \in \{0, 1\}$) to P , who computes responses ($r_i = a_{2i+c_i-1}$) and returns them. V measures round-trip times ($timer_i$) over n rounds. (3) *Authentication Phase*: V verifies proximity using $a = ss \oplus m$, the round-trip times, the allowed delay, and the speed of light. It checks $timer_i \leq 2 \times th$ and $r_i = a_{2i+c_i-1}$. If the prover is within th , the algorithm outputs 1; otherwise, 0.

Group Signature (GS): A GS allows a group member to anonymously sign a message on behalf of the group [20]. We adopt a variant of the BBS group signature scheme [17], characterized by short signatures, provable security, and high efficiency, comprising the following algorithms:

- $pp_G \leftarrow BBS.Setup(1^\lambda)$: Given the security parameter λ , it runs group parameter generation algorithm $GGen(\cdot)$ and outputs $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e(\cdot))$. Then, it obtains $g_1 \xleftarrow{\$} \mathbb{G}_1^*$, $g_2 \xleftarrow{\$} \mathbb{G}_2^*$, and $\mathbf{h}_1 \xleftarrow{\$} \mathbb{G}_1^\ell$, and returns the public parameters $pp_G \leftarrow (p, g_1, \mathbf{h}_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e(\cdot))$.
- $(sk, GK) \leftarrow BBS.KeyGen(pp_G)$: Given pp_G , it computes $x \xleftarrow{\$} \mathbb{Z}_p$, $X_2 \leftarrow g_2^x$, and outputs $(sk \leftarrow x, GK \leftarrow X_2)$.
- $\sigma \leftarrow BBS.Sign(sk = x, \mathbf{m})$: Given the message \mathbf{m} and the secret key sk , it outputs the group signature $\sigma = (A, \bar{e})$ by performing $C \leftarrow g_1 \Pi_i \mathbf{h}_1[i]^{m[i]}$, $\bar{e} \xleftarrow{\$} \mathcal{D}_{\bar{e}}$, and $A \leftarrow C^{\frac{1}{x+\bar{e}}}$.
- $\{0, 1\} \leftarrow BBS.Verify(GK, \mathbf{m}, \sigma = (A, \bar{e}))$: On input the message \mathbf{m} , signature σ , and the group public key GK , it checks $C \leftarrow g_1 \Pi_i \mathbf{h}_1[i]^{m[i]}$ and returns 1 if $e(A, g_2^{\bar{e}.vk}) = e(C, g_2)$; otherwise, returns 0.

Time-Lock Puzzle (TLP): Introduced by Rivest et al. [21], TLP encrypts messages decryptable only after a set time. We adopt the RSA-based TLP [22], leveraging non-parallelizable repeated squaring. Unlike hash-based puzzles, this prevents acceleration via multiple machines. The TLP includes:

- $\Pi \leftarrow Puzzle.Gen(1^\lambda, \kappa)$: Given the security parameter λ , this algorithm follows the same procedure as the RSA key generation [21], resulting in a private key d and its modular inverse $e = d^{-1} \pmod{\phi(n)}$. The difficulty κ is set as the number of modular squarings required, determined by $\kappa = T \cdot S$, where S is the squarings-per-second rate of a reference machine and T is the desired solving time. The value $r = 2^\kappa \pmod{\phi(n)}$ is computed, followed by the public exponent $\tilde{e} = 2^\kappa + \phi(n) - r + e$, where $z = \phi(n) - r + e$. The lower bits of \tilde{e} are composed of z , prefixed by a sequence of 0-bits and a leading 1-bit. Finally, the algorithm outputs the $pk \leftarrow \Pi = (n, \tilde{e})$ and the $sk \leftarrow \psi = (n, d)$, with the public key efficiently represented as (n, κ, z) .
- $\psi \leftarrow Puzzle.Sol(m, (n, \tilde{e}))$: Given $\Pi = (n, \tilde{e})$ and the message $0 < m < n$ chosen by the puzzle solver, it produces $c = c_1 \cdot c_2 \pmod{n}$ where $c_1 = m^{2^\kappa} \pmod{n}$ and $c_2 = m^z \pmod{n}$. Then, it sets the solution as $\psi = (m, c)$.
- $\{0, 1\} \leftarrow Sol.Verify(d, \psi)$: Using the secret key d and solution ψ , verify the correctness of $c^d \pmod{n} = m$ and return 1 if valid; otherwise, return 0.

III. FRAMEWORK ARCHITECTURE AND MODEL

System Model: Our framework model comprises five key entities: 1) *Federal Communications Commission (FCC)*: The central authority governing the SAS, responsible for establishing system parameters and enforcing regulatory compliance. 2) *Private Spectrum Databases (PSDs)*: These encompass multiple geo-location spectrum databases [1], [5] that provide real-time spectrum availability data. PSDs operate in adherence to FCC regulations, ensuring synchronization and consistency. 3) *Users*: This group includes both primary users (PUs) and secondary users (SUs) equipped with various devices (e.g., laptops, IoT, smartphones). PUs supply spectrum usage data to PSDs, while SUs query these databases for spectrum availability and CRN services. Additionally, a Nearby Device (ND) refers to any verified user within proximity. 4) *Servers*:

These are diverse network service providers (e.g., CRN, web, cloud servers) that clients access for specific services. 5) *Access Points (APs)*: Existing WiFi access points or cellular network towers in the area equipped with synchronized clocks.

Threat Model and Security Objectives: Our threat model addresses various cybersecurity attacks, focusing on privacy, anonymity, and location spoofing: (i) Users location privacy and real identity are under threat in all stages of spectrum access and CRN services due to FCC’s mandated requirement for sharing detailed coordinates, transmission data, and spectrum information. In this model, PSDs handle query responses, and CRN servers provide network services, operating as honest-but-curious entities—fulfilling their roles while attempting to infer users’ location, identity, and personal information. (ii) Users are required to provide location proofs but may act maliciously to exploit spectrum channels and services or fall victim to spoofing or compromise. Providing incorrect locations could enable access to occupied channels or unauthorized services. Potential attacks include relay attacks, distance fraud, mafia fraud, and distance hijacking [11]. (iii) During spectrum usage notifications, channel access, or CRN service requests, users may launch DoS attacks targeting PSDs or CRN servers.

Given the system and threat models, SLAP aims to achieve the following security objectives:

- **Client Privacy and Anonymity:** User location coordinates, device specifications, and personal identity remain confidential and anonymous during spectrum access, usage notifications, and CRN services, safeguarding against PSDs, CRN servers, and external attackers.
- **Location Verification and Attack Resistance:** Users are restricted from accessing spectrum data or CRN services outside their verified location. The system is resilient to distance fraud, mafia fraud, and distance hijacking, ensuring only legitimate users at authenticated locations can access services.
- **Denial-of-Service Resistance:** Spectrum channels and CRN services are safeguarded against DoS attacks, whether from users or external sources, ensuring uninterrupted and reliable service availability.

IV. THE PROPOSED SCHEME: SLAP

A. SLAP Framework Architecture and Initial Setup

Geolocation databases store frequency information and synchronize as mandated by the FCC [1]. APs within a region function as a group, each holding a pair of secret key sk_{AP} and the group verification key GK , generated by the FCC using $(sk, GK) \leftarrow BBS.KeyGen(param_G)$. To estimate a user’s physical distance, an AP performs signal strength analysis and round-trip time (RTT) measurements. Using the received signal strength (RSS), RTT, and environmental parameters (env_{params}), the algorithm $\Delta \leftarrow ProxVerif(RSS, RTT, env_{params})$ computes and outputs the estimated physical distance of the user. The FCC acts as the root issuer for credentials in the system. For a set of attributes A associated with a user’s device (e.g., device ID, type), the FCC issues Level 1 root credentials to all registered users (PUs and SUs) using the algorithm $CreateCred(L', A, sk_{FCC})$. Each user,

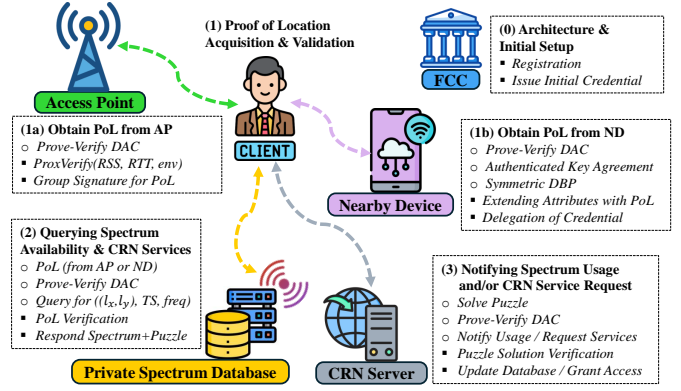


Fig. 1: SLAP Main Operations

identified by the pseudonym nym_u , obtains their credential via $GetCred(pk_{FCC}, sk_u, A) \rightarrow (cred_u, (\vec{C}, \vec{O}), dk_{L'})$. The credential $cred_u$ consists of a set commitment \vec{C} over attributes A , rooted in the FCC’s public key pk_{FCC} , the corresponding opening information \vec{O} , and a delegation key $dk_{L'}$ enabling delegation up to level L' . With this credential, the user can renew their pseudonym nym_u or delegate their credentials to another user by switching to a new public key and optionally extending the attribute set to A' . Demonstrating possession of the credential involves the user proving ownership of the secret key sk_u and generating a randomized signature over the required attributes.

B. SLAP Framework Main Operations

The flow of the SLAP framework, depicted in Fig. 1, comprises three main phases outlined as follows:

1) Proof of Location Acquisition and Validation: In this phase, the user obtains a valid proof of location (PoL) for a specified time and geographic area, with the process tailored to two complementary scenarios: densely populated areas with robust infrastructure and rural regions with limited resources.

(i) When an AP is within the user’s proximity, the user requests a PoL from the AP with the strongest signal, as outlined in Algorithm 1. Using anonymous credentials $(nym_c, cred_c)$, the user specifies attributes D , timestamp (TS) , and location coordinates (l_x, l_y) , and verifies their credentials with the FCC’s public key (Steps 1–3). Upon successful verification (Step 4), the AP evaluates proximity using signal strength and RTT measurements (Step 5). If proximity is validated (Step 6), the AP generates a group signature on the user’s location, timestamp, and credentials and transmits it to the user (Steps 7–9). The user then verifies the group signature and accepts it as valid proof of location (Steps 10–11).

(ii) In sparsely populated areas lacking WiFi APs or cellular towers, this scenario provides a fallback for obtaining location proof and anonymous credentials from nearby devices (NDs), as detailed in Algorithm 2. The client broadcasts a PoL request to NDs for the current time and location (Step 1). Upon receiving responses, the client verifies their credentials with the ND using the FCC’s public key (Steps 2–3). If valid, the ND establishes a secret session key ss via an interactive authenticated key agreement and performs a symmetric DBP

Algorithm 1 $\Phi \leftarrow \text{POL.AP}(cred_c, (l_x, l_y), TS)$

Client:

- 1: Request POL from AP
- 2: Set $D \leftarrow ((l_x, l_y), TS)$
- 3: Perform $CredProve(pk_{FCC}, sk_c, nym_c, aux_c, cred_c, D)$

Access Point:

- 4: **if** $1 \leftarrow CredVerify(pk_{FCC}, nym_c, D)$ **then**
- 5: $\Delta_c \leftarrow ProxVerify(RSS, RTT, env_{params})$
- 6: **if** $(l_x, l_y) \in \Delta_c$ **then**
- 7: Set $\mathbf{m} \leftarrow \{D, nym_c, Cred_c\}$
- 8: $\sigma_{AP} \leftarrow BBS.Sign(sk_{AP}, \mathbf{m})$
- 9: Send $(\mathbf{m}, \sigma_{AP})$ to the Client.

Client:

- 10: **if** $1 \leftarrow BBS.Verify(vk, \mathbf{m}, \sigma_{AP})$ **then**
- 11: **return** $\Phi \leftarrow (\sigma_{AP}, \mathbf{m} = (l_x, l_y, TS, nym_c, Cred_c))$

to verify the client's proximity within a threshold th (Steps 4–6). Once confirmed, the ND includes the client's location (l_x, l_y) and TS in its attributes and anonymously delegates a credential to the client with limited delegation capabilities (Steps 7–10). Using the FCC's public key and their own secret key, the client receives the delegated credential and location proof, certified within the extended attributes (Steps 11–13).

Algorithm 2 $(cred'_c, A') \leftarrow \text{POL.ND}(cred_c, (l_x, l_y), TS)$

Client:

- 1: Request POL and a delegated credential $cred'_c$ from an ND
- 2: Set $D \leftarrow ((l_x, l_y), TS)$
- 3: Perform $CredProve(pk_{FCC}, sk_c, nym_c, aux_c, cred_c, D)$

Nearby Device:

- 4: **if** $1 \leftarrow CredVerify(pk_{FCC}, nym_c, D)$ **then**
- 5: Perform $ss \leftarrow AKA(sk_{ND}, pk_{ND}, pk_c)$
- 6: **if** $1 \leftarrow SymDBP(ss, th)$ and $(l_x, l_y) \in th$ **then**
- 7: Set the new extended attributes as $A_l \leftarrow ((l_x, l_y), TS)$
- 8: Set the new delegatable key as $dk'_{L''} := \perp$
- 9: $IssueCred(pk_{FCC}, dk_{L'}, sk_{ND}, cred_{ND}, A_l, L'')$
- 10: Send $(cred'_c, dk'_{L''})$ to the client.

Client:

- 11: $(cred'_c, dk'_{L''}) \leftarrow ReceiveCred(pk_{FCC}, sk_c, A_l)$
- 12: Set $A' \leftarrow (A, \Phi)$ where $\Phi \leftarrow A_l$
- 13: **return** $(cred'_c, A')$

2) Querying Spectrum Availability and CRN Services:

Algorithm 3 details the process for querying spectrum availability, reporting spectrum usage, and accessing CRN services, primarily focusing on SUs as clients. The procedure for primary users PUs populating the database mirrors the process for querying PSDs. Given the client's location coordinates (l_x, l_y) and the current timestamp TS , the process begins with obtaining a valid POL, either from an AP or nearby devices. In areas with sufficient infrastructure, the client retrieves the proof from an AP using Algorithm 1 (Step 1) and then proves their credentials to a PSD while querying for spectrum availability or CRN services (Steps 2–4). In poorly-structured areas, the client obtains proof of location and delegated credentials from an ND using Algorithm 2 (Step 5). The delegated credential, containing the proof of location as an extended attribute, allows the client to anonymously prove their credentials to the PSD and submit queries for spectrum availability or CRN services (Steps 6–8). Notably,

clients can precompute multiple credentials offline for future use, enhancing efficiency and flexibility.

Upon receiving a query, the PSD validates the credentials and proof of location. For AP-based location proofs, the PSD verifies the group signature, while for ND-based proofs, it checks the certified attributes, including the location proof, via the underlying signature verification (Steps 9–10). Based on the request for spectrum availability or CRN services, the PSD generates a puzzle linked to the target server's public key (Steps 11, 14) and responds accordingly. While puzzle generation is included in the algorithm, PSDs typically precompute puzzles with varying difficulty levels offline, similar to spectrum data. The difficulty is determined based on the risk of DoS attacks and the server's resource capacity to manage responses. Using the device details embedded in credential attributes, the PSD distributes the tailored puzzles accordingly. Notably, the online phase of SLAP only involves proving and verifying anonymous credentials during the query process, as location proof acquisition can be completed offline in advance.

Algorithm 3 SLAP Scheme

Client:

$\rho_c \leftarrow Client.Query(cred_c, \Phi, (l_x, l_y), TS, freq):$

Give (l_x, l_y) and TS , client request POL:

- 1: **if** $\Phi \leftarrow Client.POL.AP(cred_c, (l_x, l_y), TS)$ **then**
- 2: Set $D \leftarrow ((l_x, l_y), TS, \Phi)$
- 3: Perform $CredProve(pk_{FCC}, sk_c, nym_c, aux_c, cred_c, D)$
- 4: Query a PSD for $\rho_c \leftarrow ((l_x, l_y), TS, freq)$
- 5: **elseif** $(cred'_c, \Phi) \leftarrow Client.POL.ND(cred_c, (l_x, l_y), TS)$ **then**
- 6: Given $cred_c \leftarrow cred'_c$ with $A \leftarrow (A, \Phi)$ for $((l_x, l_y), TS)$
- 7: Perform $CredProve(pk_{FCC}, sk_c, nym_c, aux_c, cred_c, A')$
- 8: Query a PSD for $\rho_c \leftarrow ((l_x, l_y), TS, freq)$

Private Spectrum Database:

$\rho_{PSD} \leftarrow PSD.Respond((l_x, l_y), TS, freq):$

- 9: **if** $1 \leftarrow CredVerify(pk_{FCC}, nym_c, D)$ **then**
- 10: **if** $1 \leftarrow BBS.Verify(vk, \mathbf{m} = (D, num_c, Cred_c), \sigma_{AP})$ **then**
- 11: Set $\Pi \leftarrow Puzzle.Gen(1^\lambda, \kappa)$ accordingly
- 12: **return** $\rho_{PSD} \leftarrow (\beta, \Pi)$ for $((l_x, l_y), TS)$
- 13: **elseif** $1 \leftarrow CredVerify(pk_{FCC}, nym_c, A')$ **then**
- 14: Set $\Pi \leftarrow Puzzle.Gen(1^\lambda, \kappa)$ accordingly
- 15: **return** $\rho_{PSD} \leftarrow (\beta, \Pi)$ for $((l_x, l_y), TS)$

Client: Notifying spectrum usage to PSDs or accessing services:

- 16: **for** Spectrum usage data or service requests as m **do**
- 17: Given $\Pi \leftarrow (n, \tilde{e})$ for the PSD or the target server
- 18: Perform $\psi \leftarrow Puzzle.Sol(m, (n, \tilde{e}))$
- 19: Perform $CredProve(pk_{FCC}, sk_c, nym_c, aux_c, cred_c, D)$
- 20: Sends (m, ψ) to the PSD or the CRN server

Private Spectrum Database/CRN Server:

- 21: **if** $1 \leftarrow CredVerify(pk_{FCC}, nym_c, D)$ **then**
- 22: **if** $1 \leftarrow Sol.Verify(m, c, d)$ **then**
- 23: PSD/CRN Server **return** 1, and update DB or grant access

3) Notifying Spectrum Usage and/or CRN Service Request: To report spectrum usage data or access CRN services, users must solve the puzzle previously obtained, tied to the target server's public key. Given a message m , representing spectrum usage data or an access request, the user computes the puzzle solution via repeated squaring and submits it alongside proof of their anonymous credentials to the PSD

or CRN server (Steps 16–20). Upon receiving the message and solution, the server validates the anonymous credentials (Step 21) and verifies the puzzle solution (Step 22). If both are verified, the PSD updates its database, or the CRN server grants access to the requested resources. Unlike other schemes, spectrum usage notifications also leverage anonymous credentials with attributes, improving frequency information quality while adhering to FCC coexistence requirements.

V. SECURITY ANALYSIS

We provide security proofs addressing the threat model:

Lemma 1. *SLAP ensures anonymous user authentication by leveraging the strong anonymity, soundness, and unforgeability properties of the ZKPoK and SPSEQ-UC signature schemes.*

Proof. SLAP ensures robust anonymity, preventing any entity from tracing or inferring user identity or information beyond the required credentials during both issuance/delegation and presentation phases. Malicious verifiers cannot differentiate between users, and this strong anonymity is achieved without relying on a trusted setup. The framework’s anonymity is grounded in the knowledge soundness of Zero-Knowledge Proof of Knowledge (ZKPoK), the Decisional Diffie-Hellman (DDH) assumption, and the SPSEQ-UC scheme [23], collectively ensuring origin-hiding, conversion-privacy, and derivation-privacy [14]. Origin-hiding guarantees indistinguishability of randomized signatures; derivation-privacy ensures extended commitment vectors remain indistinguishable; and conversion-privacy ensures new signatures generated with switched user keys are indistinguishable from fresh signatures. These privacy properties can be repeatedly applied in any order without compromising security. \square

Corollary 1. *SLAP provides location privacy for the spectrum access via the unlikability of the credentials formed on the signature and commitment pairs.*

Proof. The location privacy of SLAP is ensured by the unlinkability of signature-commitment pairs generated using the SPSEQ-UC scheme [14]. This unlinkability is achieved through signature re-randomization and user public key switching, enabling the repeated disclosure of the same commitment-signature pair without linkability. Provided no identifying attributes are included, newly generated signatures are indistinguishable from the originals. This property, formally proven secure under the group model, ensures that credential presentations remain unlinkable to verifiers. \square

Lemma 2. *The SLAP framework ensures location verification of the users during spectrum access and queries via (i) the unforgeability of the group signatures and enhanced signal strength measurements; (ii) public key distance-bounding protocol and anonymous delegation of credentials.*

Proof. In the first scenario, the risk of fraud against the AP is negligible due to robust security measures. Connection to the AP is secured using a broadcasted sequence number transmitted within a short time window (e.g., 100–500 ms), mitigating potential attacks [24], while proximity is validated through signal strength measurements. The AP’s group signature on the *cred*, *POL*, and *TS* verifies that the user is within the

AP’s coverage area. The unforgeability of the \mathcal{GS} , grounded in the q-SDH assumption and supported by a tighter security proof in the algebraic group model, ensures the integrity of the location verification provided to the PSD [17]. Additionally, the location proof is non-transferable, as it is cryptographically bound to the current *TS* and the user’s verified credentials.

In the ND scenario, location verification is ensured through the following mechanisms: (i) The security of the AKA protocol, based on the hardness of the Diffie-Hellman and discrete logarithm problems in the random oracle model [18]. (ii) The negligible failure probability of the symmetric \mathcal{DBP} [19]. Specifically, in the canonical OTDB scheme [18], with $m \in \{0, 1\}^{2n}$ during initialization, the optimal probability for an adversary to correctly respond to all challenges is $(\frac{3}{4})^n$, providing strong resistance to distance fraud, mafia fraud, and distance hijacking [18]. (iii) The unforgeability and anonymity of \mathcal{DAC} delegation. An adversary attempting to forge a new delegated credential with another user’s certified *POL* must either forge the SPSEQ-UC scheme or compromise the NIZK proof scheme, both of which are provably secure [14]. \square

Corollary 2. *SLAP offers a counter-DoS mechanism for spectrum access, usage notification, and obtaining CRN services via public-key time-lock puzzles.*

Proof. The security of the TLP is directly grounded in Rivest’s construction [21], which relies on the hardness of the integer factorization problem and the computational properties of modular exponentiation with a power-of-two exponent. Specifically, deriving c without performing κ modular exponentiation operations during puzzle-solving (step 18 of Algorithm 3) is computationally infeasible for an adversary. Furthermore, reducing \tilde{e} to e and computing $\phi(n)$ is provable as hard as factoring n into its two large prime factors. To maintain security, the PSD must avoid disclosing multiple \tilde{e} values associated with the same key pair, as such disclosure would enable efficient factorization of n and compromise the scheme. \square

VI. PERFORMANCE EVALUATION

A. Metrics, Selection Rationale, and Configurations

Evaluation Metrics and Rationale: We conduct analytical and empirical evaluations of the SLAP framework, assessing its computational costs and communication overhead across all phases and employed primitives, including \mathcal{DAC} , \mathcal{GS} , \mathcal{DBP} , and \mathcal{TLP} . As no existing solutions offer a similarly comprehensive set of features, a direct performance comparison is not feasible. Instead, we provide a detailed performance analysis of SLAP across key metrics to evaluate its feasibility and practicality. Additionally, we present a qualitative and analytical comparison with selected schemes addressing subsets of these features in the context of spectrum query to SAS. The evaluation is structured as follows.

Hardware, Software Libraries, and Parameters: Our experiments were conducted on a desktop with an 11th Gen Intel Core *i9-11900K@3.50 GHz*, 64 GiB RAM, 1 TB SSD, running Ubuntu 22.04.4 LTS. The implementation utilized libraries and tools such as *DAC-from-EQS*¹, *bbs-node ref*

¹<https://github.com/mir-omid/DAC-from-EQS>

Phase	Entity	Analytical Computational Cost	Empirical Cost	Communication Overhead
PoL.AP	Client	$((k+11)\mathbb{G}_1 + 3\mathbb{G}_2 + \mathbb{G}_2^2) + \mathbb{G}_1^{ D } + 1P + 3\mathbb{G}_T + (\sum_{i=1}^{ D }(\mathbb{G}_1^{u_i} + \mathbb{G}_1))$	20.17 ms	$((k+8) \mathbb{G}_1 + 2 \mathbb{G}_2 + 3 \mathbb{Z}_p)$
	AP	$2E^k + E^2 + 5E + \mathbb{G}_2^{ S } + 3\mathbb{G}_T + 9\mathbb{G}_1 + O(1) + \sum_{i=1}^{ D }(\mathbb{G}_2^{ S-d_i } + \mathbb{G}_2)$	61.26 ms	$ TS + (l_x, l_y) = 2008B$
PoL.ND	Client	$((k+3)\mathbb{G}_1 + \mathbb{G}_2 + \mathbb{G}_2^2) + E_M + H + rnd + O(1) + (\mathbb{G}_1^{ D }) + (\sum_{i=1}^{ D }(\mathbb{G}_1^{u_i} + \mathbb{G}_1))$	31.75 ms	$(3k+8) \mathbb{G}_1 + 4 \mathbb{G}_2 $
	ND	$2E^k + E^2 + 5E + \mathbb{G}_2^{ S } + 3\mathbb{G}_1^2 + 2\mathbb{G}_1^n + \mathbb{G}_2^2 + (k+5)\mathbb{G}_1 + \mathbb{G}_2 + \sum_{i=1}^{ D }(\mathbb{G}_2^{ S-d_i } + \mathbb{G}_2) + E_M + H + rnd + O(1)$	78.05 ms	$+ TS + (l_x, l_y) + (k+1) \mathbb{Z}_p = 1856B$
Spectrum Query	Client	$((k+3)\mathbb{G}_1 + \mathbb{G}_2 + \mathbb{G}_2^2) + (\mathbb{G}_1^{ D }) + (\sum_{i=1}^{ D }(\mathbb{G}_1^{u_i} + \mathbb{G}_1)) + O(1)$	17.22 ms	$(k+5) \mathbb{G}_1 + \mathbb{G}_2 + \mathbb{Z}_p +$
	PSD	$2E^k + E^2 + 5E + \mathbb{G}_2^{ S } + 1P + 3\mathbb{G}_T + 2\mathbb{G}_2 + 8\mathbb{G}_1 + O(1) + \sum_{i=1}^{ D }(\mathbb{G}_2^{ S-d_i } + \mathbb{G}_2)$	61.39 ms	$ TS + (l_x, l_y) + \beta = 3080B$
Notify/Service Request	Client	$\kappa \times S_q + E_M + (k+3)\mathbb{G}_1 + 2\mathbb{G}_2 + \mathbb{G}_1^2 + (\mathbb{G}_1^{ D }) + (\sum_{i=1}^{ D }(\mathbb{G}_1^{u_i} + \mathbb{G}_1))$	17.22 ms + $\kappa \times S_q$	$(k+5) \mathbb{G}_1 + \mathbb{G}_2 + \mathbb{Z}_p $ $ m + TS + \Pi + \psi $
	PSD/	$2E^k + E^2 + 5E + \mathbb{G}_2^{ S } + \mathbb{G}_2 + \sum_{i=1}^{ D }(\mathbb{G}_2^{ S-d_i } + \mathbb{G}_2)$	59.01 ms	$= 2304B$

Computations: \mathbb{G}_1 , \mathbb{G}_2 , and \mathbb{G}_T denote exponentiation in the respective groups. E^k represents a k -pairing product, where $k = 1$ corresponds to a single pairing operation; P denotes pairing over the BN-256 curve. E_M represents modular multiplication ($n = 2048$). rnd denotes random string selection. H is a cryptographically secure hash function (SHA-256), and S_q represents the repeated squaring time to solve a puzzle. $O(1)$ signifies signal transmission and internet communication time, typically in the microsecond range. Let $D = (d_i)_{i \in [k]}$ and $S = \bigcup_i d_i$ for all $i \in [k]$, where k is the delegation level ($L = 2$ in this scheme), and (d_i, u_i) denotes disclosed and undisclosed attributes at level i . κ represents puzzle difficulty. **Communication:** Bits and bytes are denoted by b and B , respectively. Group sizes are $|\mathbb{G}_1| = |\mathbb{Z}_p| = 256b$, $|\mathbb{G}_2| = 512b$, and $|\mathbb{G}_T| = 3072b$, with modular arithmetic over $n = 2048$. Messages $|m| < 256B$, timestamps $|TS|$ are $8B$ (on a 64-bit Unix system), high-precision location coordinates are $16B$, and spectrum availability information $|\beta|$ (based on FCC raw data) is approximately $560 B$.

TABLE I: Computational Costs and Communication Overhead of SLAP Framework

erence², time-lock-puzzle³, and OpenSSL⁴. These were used for implementing cryptographic primitives, including hash functions, modular arithmetic, exponentiation, and core SLAP components. The setup included SHA-256 for hashing, [23] for set commitments, [14] for SPSEQ-UC, BN256 curve for binding and ECC, Schnorr-style ZKP with Damgard's technique for DAC [14], and NIZKs derived via the Fiat-Shamir heuristic, achieving approximately 100-bit security.

B. Experimental Results

The analytical and empirical evaluation of cryptographic overhead, computational costs, and communication overhead for each phase of the SLAP framework is summarized in TABLE I and detailed below:

Cryptographic Overhead: To prove a credential, the user randomizes their *cred* and *nym* and employs a ZKPoK on the secret key *sk* and randomness *aux* to generate a new randomized *nym* along with a subset of attributes D using a set commitment scheme. Signature conversion, signature representation adjustment, and adaptation for a new set commitment take approximately $2ms$, $5ms$, and $13ms$, respectively. On commodity hardware, solving puzzles at difficulty levels κ (number of squarings) set to 10^3 , 15×10^3 , 50×10^3 , 10^5 , and 10^6 requires $3.9ms$, $56.31ms$, $194ms$, $784ms$, and $3.786s$, respectively. Verifying a puzzle solution, which involves RSA decryption (modular exponentiation), takes about $797\mu s$. Group signing and verification are completed in $2.26ms$ and $3.17ms$, respectively, with batch verification reducing costs on the PSD side. In the employed DBP , the AKA requires one ECC multiplication ($0.612ms$), one hashing ($0.35ms$), and random string selection ($0.045ms$). Rapid bit exchange occurs on a nanosecond scale, with a δ distance fraud probability corresponding to changes around $100cm$, negligible compared to other protocol aspects. For $ProxVerify()$, performed by the AP using signal strength and RTT techniques, the process is considered to take approximately 1-10ms.

Computational Costs: (i) *PoL.AP Phase:* The client proves anonymity and verifies the group signature, while the AP validates credentials, executes the *ProxVerify* algorithm, and generates a group signature for the location proof. (ii) *PoL.ND Phase:* This phase involves interactive protocols between two users, including credential proof and verification, key agreement, symmetric DBP , and credential delegation/receipt with the location proof as a certified attribute. (iii) *Spectrum Query Phase:* The user submits a valid location proof (from the AP or ND) when querying a PSD for spectrum availability at specific coordinates and timestamps. The PSD verifies credentials, checks the location proof, and provides spectrum information along with a public key puzzle tailored to the user's attributes. (iv) *Notifying Spectrum Usage or Requesting CRN Services Phase:* The user solves the PSD or CRN server's public key puzzle, proves credentials, and submits the solution. The PSD/server verifies the solution before granting access to services or updating the database.

Communication Overhead: The communication complexity and data sizes for each phase are summarized in TABLE I. In our scheme, all attributes are assumed to have uniform size. The credential includes $|cred| + |sk| + |nym|$ within the set commitment and SPSEQ-UC schemes, maintaining a constant size independent of the number of attributes, calculated as $4|\mathbb{G}_1| + |\mathbb{G}_2| + |\mathbb{Z}_p|$, resulting in a credential size of $1792 b$. The size of \bar{C} corresponds to the delegation level ($L = 2$), with communication complexity increasing linearly with the number of attributes and delegations. Using publicly available raw database data from the FCC⁵, we estimated each database block to contain approximately 560 bytes of information, supplemented with synthetic data for evaluation purposes.

Comparison with SOTA: We perform a qualitative and analytical comparison of the achieved features with other state-of-the-art location privacy schemes, as detailed in TABLE II. For a fair evaluation, we consider spectrum query costs from both the client's and PSD's perspectives, system communication

²<https://github.com/microsoft/bbs-node-reference/tree/main>

³<https://github.com/pmuens/time-lock-puzzle>

⁴<https://www.openssl.org/>

⁵<https://enterpriseefiling.fcc.gov/dataentry/public/tv/lmsDatabase.html>

Scheme	Features					Delay				Total Communication
	Setting	Loc.Privacy	Anonym	Loc.Verification	Counter-DoS	SU	PSD	E2E	PoL	
Troja et al [25]	1-DB	Peer-to-Peer	✓	✗	✗	1650 ms	11760 ms	13410 ms	✗	12 MB
Li et al [12]	1-DB	Pseudo-ID	✗	WiFi AP+Loc.Server	✗	✗	✗	✗	210 ms	✗
Xin et al [7]	1-DB	PIR	✗	WiFi AP+QRA	✗	292.8 ms	142.7 ms	407.4 ms	430.1 ms	325KB
LP-Chor [16]	ℓ-DB	PIR	✗	✗	✗	7.7 ms	480 ms	620 ms	✗	753 KB
LP-Goldberg [16]	ℓ-DB	PIR	✗	✗	✗	320 ms	1210 ms	1780 ms	✗	6 MB
RAID-LP-Chor [16]	ℓ-DB	PIR	✗	✗	✗	0.4 ms	22 ms	210 ms	✗	125 KB
Zeng et al [26]	1-DB	BS+ECC	PseudoID	✗	✗	87 ms	27 ms	135 ms	✗	1.24 KB
TrustSAS [5]	ℓ-DB	PIR	EPID	✗	✗	329.4 ms	324.6 ms	4954 ms	✗	1.25 MB
PACDoSQ [6]	ℓ-DB	PIR	Tor	✗	HBP	28.1 ms	199 ms	1373.6 ms	✗	605.92 KB
SLAP	1-DB	D.AC	D.AC	WiFi AP+G.Sig DBP+D.AC	TLP	17.22 ms	61.39 ms	78.61 ms	107.17 ms 109.8 ms	3.08KB

Libraries: Virtual Machines running Ubuntu simulated PIR costs, using the *percy++* library⁶ for multi-server PIR, the *Open Quantum-Safe* library⁷ for PQC primitives, and *OpenSSL* for cryptographic operations and arithmetic. **Variables:** We consider six databases for multi-DB schemes with $|DB| = 560MB$ and 400 rows/columns as described in [7]. Key terms include *BS* (base station), *HBP* (hash-based puzzles), *G.Sig* (group signature), *QRA* (quadratic residue assumption), and *EPID* (enhanced privacy ID based on direct anonymous attestation).

TABLE II: Qualitative and Analytical Comparison with Existing Location Privacy Schemes

overhead, and end-to-end delay for retrieving a single block from the geo-location databases as a measure of scalability. As shown in TABLE II, our approach delivers all necessary features for secure, location-private, and anonymous spectrum access, while offering architecture-flexible and efficient location verification with the lowest end-to-end delay and minimal communication burden on the system.

VII. CONCLUSION

The increasing demand for communication resources has driven the development of SAS, but regulatory requirements for disclosing sensitive user data raise privacy and security concerns, including the need for robust location verification and resilience against DoS attacks. To address these challenges, we proposed SLAP, a framework ensuring strong location privacy, full anonymity during spectrum queries, and adaptive dual-scenario location verification while integrating TLP-based counter-DoS mechanisms. We formally proved its security and demonstrated its efficiency and scalability through comprehensive evaluations.

REFERENCES

- [1] P. Agarwal, M. Manekiya, and T. Ahmad, "A survey on citizens broadband radio service (cbrs)," *Electronics*, vol. 11, no. 23, 2022.
- [2] S. Shi, Y. Xiao, W. Lou, C. Wang, X. Li, Y. T. Hou, and J. H. Reed, "Challenges and new directions in securing spectrum access systems," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6498–6518, 2021.
- [3] N. Nguyen-Thanh, D.-T. Ta, and V.-T. Nguyen, "Spoofing attack and surveillance game in geo-location database driven spectrum sharing," *IET Communications*, vol. 13, no. 1, pp. 74–84, 2019.
- [4] G. Jakimoski and K. Subbalakshmi, "Denial-of-service attacks on dynamic spectrum access networks," in *IEEE International Conference on Communications Workshops*. IEEE, 2008, pp. 524–528.
- [5] M. Grissa, A. A. Yavuz, B. Hamdaoui, and C. Tirupathi, "Anonymous dynamic spectrum access and sharing mechanisms for the cbrs band," *IEEE Access*, vol. 9, pp. 33 860–33 879, 2021.
- [6] S. Darzi and A. A. Yavuz, "Privacy-preserving and post-quantum counter denial of service framework for wireless networks," in *IEEE Military Communications Conference (MILCOM)*. IEEE, 2024, pp. 1076–1081.
- [7] J. Xin, M. Li, C. Luo, and P. Li, "Privacy-preserving spectrum query with location proofs in database-driven crns," in *2016 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2016, pp. 1–6.
- [8] L. Zhang, C. Fang, Y. Li, H. Zhu, and M. Dong, "Optimal strategies for defending location inference attack in database-driven crns," in *2015 IEEE International Conference on Communications (ICC)*.
- [9] R. Zhu, L. Xu, Y. Zeng, and X. Yi, "Lightweight privacy preservation for securing large-scale database-driven cognitive radio networks with location verification," *Security and Communication Networks*, 2019.
- [10] M. Ul Hassan, M. H. Rehmani, M. Rehan, and J. Chen, "Differential privacy in cognitive radio networks: a comprehensive survey," *Cognitive Computation*, vol. 14, no. 2, pp. 475–510, 2022.
- [11] K. Zeng, S. K. Ramesh, and Y. Yang, "Location spoofing attack and its countermeasures in database-driven cognitive radio networks," in *2014 IEEE Conference on Communications and Network Security*, 2014.
- [12] Y. Li, L. Zhou, H. Zhu, and L. Sun, "Privacy-preserving location proof for securing large-scale database-driven cognitive radio networks," *IEEE Internet of Things Journal*, vol. 3, no. 4, pp. 563–571, 2015.
- [13] S. Darzi and A. A. Yavuz, "Counter denial of service for next-generation networks within the artificial intelligence and post-quantum era," in *IEEE 6th International Conference on Trust, Privacy and Security in Intelligent Systems, and Applications*. IEEE, 2024, pp. 138–147.
- [14] O. Mir, D. Slamanig, B. Bauer, and R. Mayrhofer, "Practical delegatable anonymous credentials from equivalence class signatures," *Proceedings on Privacy Enhancing Technologies*, 2023.
- [15] M. Grissa, A. Yavuz, and B. Hamdaoui, "Trustsas: A trustworthy spectrum access system for the 3.5 ghz cbrs band," in *IEEE INFOCOM Conf. on Computer Communications*. IEEE, 2019, pp. 1495–1503.
- [16] M. Grissa and A. A. Yavuz, "Location privacy in cognitive radios with multi-server private information retrieval," *IEEE Transactions on Cognitive Communications and Networking*, 2019.
- [17] S. Tessaro and C. Zhu, "Revisiting bbs signatures," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2023, pp. 691–721.
- [18] H. Kılınc and S. Vaudenay, "Efficient public-key distance bounding protocol," in *22nd International Conference on the Theory and Application of Cryptology and Information Security*. Springer, pp. 873–901.
- [19] S. Vaudenay, "Private and secure public-key distance bounding: application to nfc payment," in *International Conference on Financial Cryptography and Data Security*. Springer, 2015, pp. 207–216.
- [20] A. A. Yavuz, K. Sedghighadikolaei, S. Darzi, and S. E. Nouma, "Beyond basic trust: Envisioning the future of nextgen networked systems and digital signatures," in *5th IEEE Intern. Conf. on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*. IEEE, 2023.
- [21] R. L. Rivest, A. Shamir, and D. A. Wagner, "Time-lock puzzles and timed-release crypto," 1996.
- [22] Y. I. Jerschow and M. Mauve, "Offline submission with rsa time-lock puzzles," in *2010 10th IEEE International Conference on Computer and Information Technology*. IEEE, 2010, pp. 1058–1064.
- [23] G. Fuchsbaauer, C. Hanser, and D. Slamanig, "Structure-preserving signatures on equivalence classes and constant-size anonymous credentials," *Journal of Cryptology*, vol. 32, pp. 498–546, 2019.
- [24] S. Saroiu and A. Wolman, "Enabling new mobile applications with location proofs," in *Proceedings of the 10th workshop on Mobile Computing Systems and Applications*, 2009, pp. 1–6.
- [25] E. Troja and S. Bakiras, "Leveraging p2p interactions for efficient location privacy in database-driven dynamic spectrum access," in *22nd ACM SIGSPATIA Conference*, 2014, pp. 453–456.
- [26] Y. Zeng, L. Xu, X. Yang, and X. Yi, "An efficient privacy-preserving protocol for database-driven cognitive radio networks," *Ad hoc networks*, vol. 90, p. 101739, 2019.