SZ-SAS: A Framework for Preserving Incumbent User Privacy in SAS-Based DSA Systems

Douglas Zabransky(✉), He Li, Chang Lu, and Yaling Yang

Virginia Polytechnic Institute and State University, Blacksburg, VA 24060, USA
{dmz5e,heli,changl7,yyang8}@vt.edu

Abstract. Dynamic Spectrum Access (DSA) is a promising solution to alleviate spectrum crowding. However, geolocation database-driven spectrum access system (SAS) presents privacy risks, as sensitive Incumbent User (IU) operation parameters are required to be stored by SAS in order to perform spectrum assignments properly. These sensitive operation parameters may potentially be compromised if SAS is the target of a cyber attack or SU inference attack. In this paper, we propose a novel privacy-preserving SAS-based DSA framework, Suspicion Zone SAS (SZ-SAS). This is the first framework which protects against both the scenario of inference attacks in an area with sparsely distributed IUs and the scenario of untrusted or compromised SAS. Evaluation results show SZ-SAS is capable of utilizing compatible obfuscation schemes to prevent the SU inference attack, while operating using only homomorphically encrypted IU operation parameters.

Keywords: Dynamic Spectrum Access · Inference attack
Location privacy

1 Introduction

Dynamic Spectrum Allocation (DSA) allows Secondary Users (SUs) to transmit opportunistically on underutilized spectrum while avoiding the creation of interference which would impact the operation of the legacy users, known as Incumbent Users (IUs).

Because the IUs in bands of interest are comprised of federal government and military systems, the operational security (OPSEC) of these users is of paramount importance. The authors of [4] identify several operational attributes of these systems which should remain confidential, including geolocation, transmit protection contours, and times of operation. However, current DSA designs include exclusion zone (E-Zone) based spectrum access systems (SAS), in which SAS is a database containing plaintext information allowing the determination of regions (E-Zones) in which SUs are not permitted to operate because they will

I. Moerman et al. (Eds.): CROWNCOM 2018, LNICST 261, pp. 78–88, 2019.
https://doi.org/10.1007/978-3-030-05490-8_8

create harmful interference to IUs. The authors of [3] first addressed this security risk by proposing a SAS framework which utilizes homomorphic encryption in order to prevent SAS from directly accessing IU operation parameters.

However, the framework proposed in [3] is susceptible to inference attacks, as it has no method of determining queries which could indicate the execution of an inference attack and no method of obfuscating responses to these queries. The SU inference attack, first defined in [2], allows adversarial SUs to correlate the results of seemingly innocuous queries in order to infer the geolocation or transmit protection contour information of an IU by simply observing which areas are available for transmission and which areas fall within the boundaries of an E-Zone. The most effective proposed countermeasure against this attack is the introduction of obfuscation [2,6]. However, the proposed obfuscation strategies require SAS to have intimate knowledge of IU operation parameters, rendering them incompatible with the secure SAS designs proposed in [3]. Additionally, many of these obfuscation techniques are not particularly effective in the case of sparse IUs, as they either rely upon grouping nearby IUs or result in obfuscated E-Zones which maintain the same geolocation center as their unobfuscated counterparts. We address this gap in the literature with our contributions described in the remainder of this paper.

Our contributions can be summarized as follows:

- We propose a novel database-driven DSA framework, Suspicion Zone SAS (SZ-SAS), the first such framework which allows for obfuscation to be applied on a per-user or per-group basis based upon the query history of an individual user and the first such framework which protects against both the scenario of inference attacks in an area with sparsely distributed IUs and the scenario of untrusted SAS.
- We introduce a modified inference attack, showing a lower bound of privacy provided by non-obfuscated SAS responses than previously suspected.
- We provide and analyze multiple obfuscation techniques which are compatible with the proposed DSA framework.

The rest of this paper is structured as follows: Section 2 introduces SZ-SAS and the cryptographic background upon which it is built, Sect. 3 discusses the problem of inference attacks and techniques which could be employed in SZ-SAS to prevent such attacks, and Sect. 4 concludes the paper.

2 SZ-SAS Background and System Model

From a high-level view, SZ-SAS leverages homomorphic proxy re-encryption to encrypt operation parameters from IUs such that SAS has no direct knowledge of these parameters. Utilizing the homomorphic nature of the chosen cryptosystem, SZ-SAS uses a novel method for maintaining and utilizing an encrypted count of potentially suspicious queries made by each SU. Then, it restricts SUs which have exceeded a given IU's specified threshold of suspicious queries from querying the IU's actual E-Zone. Instead, it calculates their query results from obfuscated E-Zones, which hide the true geolocation of the IU in question.

In this section, we first discuss the cryptographic basis of our framework. We then introduce the details of SZ-SAS framework and define its operations.

2.1 Cryptographic Preliminaries

SZ-SAS utilizes the AFGH cryptosystem [1], which is a single-hop, unidirectional, homomorphic proxy re-encryption scheme based upon bilinear maps. In this section, we discuss the basis of this cryptosystem and the cryptographic assumptions upon which it was designed.

The AFGH Homomorphic Proxy Re-encryption Scheme. The AFGH cryptosystem was chosen based upon several criteria, primary of which are functionality and overhead. SZ-SAS requires a cryptosystem in which SAS can perform operations securely on encrypted parameters. Because SZ-SAS will potentially be processing thousands of queries per minute and thus will need to reduce its computation and communication overhead, the relatively light-weight partially homomorphic AFGH scheme was chosen.

AFGH as proposed in [1] is homomorphic with respect to multiplication. However, our scheme requires an accumulator and so we must modify AFGH to become homomorphic with respect to addition. In order to accomplish this, we simply exponentiate the generator, Z of G_T, by our plaintext prior to encryption as Z^{PT}. This will allow us to perform addition, but the discrete log problem (DLP) will need to be solved in order to recover the actual plaintext. Thus, we have built our system to operate without requiring the plaintext to be recovered from Z^{PT} in order to avoid the computations required to solve the DLP. We are also able to perform multiplication of a ciphertext by a plaintext value using exponentiation. We define the encrypted addition operation and the plaintext multiplication created by our modification as $\oplus$ and $\otimes$ for the remainder of this paper and describe the construction and important features of the additive AFGH cryptosystem below:

- **System Parameters:** $e : G \times G \rightarrow G_T$ is a Type 1 bilinear map, g is a random generator of G, and $Z = e(g, g)$ is a random generator of G_T.
- **Key Generation:** Set a group secret key $SK_a \xleftarrow{\$} Z_p^*$ and public key $PK_a = g^{SK_a}$ for all IUs.
- **Re-encryption Key Generation:** To re-encrypt a level 2 ciphertext which was originally encrypted with IUs' public key into a level 1 ciphertext which can be decrypted by SU b's secret key, a re-encryption key must be generated. This key is generated with SU b's public key, PK_b, and IUs' private key, SK_a, as $RK_{a\rightarrow b} = PK_b^{1/SK_a}$. This is equivalent to g^{SK_b/SK_a}.

2.2 System Model, Operations, and Correctness

The structure of SZ-SAS is depicted in Fig. 1 and is comprised of a 4-party SAS structure consisting of key manager, IUs, SAS, and SUs.

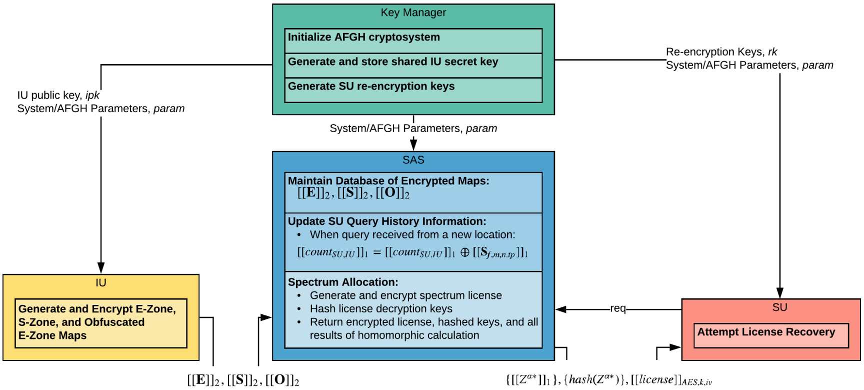

Key Manager
Initialize AFGH cryptosystem
Generate and store shared IU secret key
Generate SU re-encryption keys
IU public key, *ipk*
System/AFGH Parameters, *param*
Re-encryption Keys, *rk*
System/AFGH Parameters, *param*
System/AFGH Parameters, *param*
SAS
Maintain Database of Encrypted Maps:
$[[\mathbf{E}]]_2, [[\mathbf{S}]]_2, [[\mathbf{O}]]_2$
Update SU Query History Information:
When query received from a new location:
$[[count_{SU,IU}]]_1 = [[count_{SU,IU}]]_1 \oplus [[\mathbf{S}_{f,m,n,tp}]]_1$
Spectrum Allocation:
Generate and encrypt spectrum license
Hash license decryption keys
Return encrypted license, hashed keys, and all results of homomorphic calculation
IU
Generate and Encrypt E-Zone, S-Zone, and Obfuscated E-Zone Maps
req
SU
Attempt License Recovery
$[[\mathbf{E}]]_2, [[\mathbf{S}]]_2, [[\mathbf{O}]]_2$
$\{[[Z^{\alpha *}]]_1\}, \{hash(Z^{\alpha *})\}, [[license]]_{AES,k,iv}$

Fig. 1. SZ-SAS system framework overview

Key Manager Operations. The key manager generates re-encryption keys for each SU during the SU's initial registration with SAS. The key manager should either be controlled by a government entity or trusted-third party as re-encryption key generation requires knowledge of the shared IU secret key.

IU Operations. Each IU is responsible for the generation of its E-Zone, Suspicion Zone (S-Zone), and obfuscated E-Zone maps. Each of these maps is represented by an $M \times N$ matrix and contains information for the entire area covered by SAS. A series of E-Zone maps, each represented by a matrix $\mathbf{E}$, are generated via a chosen path loss model and a set of discretized SU maximum transmit power levels, (TP). Any grid location for which the path loss value exceeds the interference threshold of the IU is considered part of the E-Zone. If channel f at grid location (m, n) and transmit power level (tp) is considered to be E-Zone by the IU, then the value of the E-Zone map $\mathbf{E}_{f,m,n,tp}$ is a random non-zero element picked from Z_p. Otherwise, $\mathbf{E}_{f,m,n,tp}$ is set to 0.

Because the area around the protection contour provides the most information to a SU performing an inference attack, the IU defines the cells adjacent to the protection contour as S-Zone cells and generates a series of maps, represented by matrix $\mathbf{S}$, corresponding to the contour of each E-Zone map. If channel f at grid location m, n and and transmit power level (TP) is considered to be S-Zone by the IU, then the value of the S-Zone map $\mathbf{S}_{f,m,n,tp}$ is set to 1. Otherwise, $\mathbf{S}_{f,m,n,tp}$ is set to 0.

Obfuscated E-Zone maps essentially are distorted and enlarged E-Zones and are generated using an obfuscation scheme as described in Sect. 3.3. These maps, represented by matrix $\mathbf{O}$, follow the same value assignment rules as E-Zone maps. The IU also determines a suspicious request threshold, represented by τ,which is the number of queries an SU is allowed to make from S-Zone areas.

Each IU prepares these three categories of maps, encrypts each map value with the IU public key, and commits the resulting encrypted maps and the unencrypted threshold value to SAS. The encryption of these maps is essential as SAS can potentially be compromised and these maps may be used to derive sensitive parameters, such as the geolocation and operation times of the IU.

SAS Operations. SAS is responsible for performing three main operations, maintaining the database of all encrypted maps, $[\![\mathbf{E}]\!]$, $[\![\mathbf{S}]\!]$, $[\![\mathbf{O}]\!]$, updating the encrypted number of suspicious queries made by each SU for each IU, represented by $[\![count_{SU,IU}]\!]$, and performing spectrum assignment computations in response to SU requests.

SAS updates $[\![count_{SU,IU}]\!]$ whenever a query is initially received from an SU which is querying from a new location. When a query is received from location (m, n) for channel f at transmit power level tp, SAS first determines if this is the querying SU's first query from this location. If it is, SAS updates $[\![count_{SU,IU}]\!] = [\![count_{SU,IU}]\!] \oplus [\![\mathbf{S}_{f,m,n,tp}]\!]$. Otherwise, SAS does not update $[\![count_{SU,IU}]\!]$. This allows SAS to maintain an encrypted count of queries originating from grid locations which have been designated as potentially suspicious due to the potential information revealed in a SAS response to an SU performing an inference attack from this location.

The spectrum assignment computation is the primary functionality of SAS. SAS must generate a spectrum license consisting of the SU's access information (expiration time, transmit power level, location, etc.) and a digital signature over this access information. This license will be transmitted to the SU in two possible situations. In the first situation, the SU has not exceeded the threshold of suspicious requests (i.e. $count_{SU,IU} < \tau$) and is not located in the E-Zone of any IU. In the second situation, the threshold has been exceeded and, thus, the SU must not be in the obfuscated E-Zone.

Because SAS does not have direct knowledge of $[\![\mathbf{E}]\!]$, $[\![\mathbf{S}]\!]$, $[\![\mathbf{O}]\!]$, it cannot directly determine whether the SU is located within an IU's E-Zone or obfuscated E-Zone. Therefore, the assignment is performed via a special process which leverages the properties of AFGH. First, SAS updates $[\![count_{SU,IU}]\!] = [\![count_{SU,IU}]\!] \oplus [\![\mathbf{S}_{f,m,n,tp}]\!]$ as described previously. Next, SAS generates the spectrum license, which must be encrypted in a manner such that it may only be decrypted if it is a valid spectrum request, as described previously. In order to accomplish this, SAS utilizes a cascade encryption scheme and a specialized homomorphic calculation as described in the following paragraph.

First, SAS generates a separate symmetric encryption key for each IU by selecting a random element α in G_T, and uses this element to exponentiate the generator, Z, resulting in Z^α. Then, SAS hashes Z^α with a cryptographic hash function and the resulting hash digest is split into two bit strings, k and iv. The hash function used in this step will be referred to as the primary hash function for the remainder of this paper. Each IU's k and iv bit strings are then used as the secret key and initialization vector for a block cipher operating in a stream-like mode, such as CTR mode AES, and the license is sequentially encrypted by each k and iv pair via cascade encryption, resulting in $[\![license]\!]_{AES,k,iv}$. Each IU's Z^α and a series of threshold values from 0 to τ, denoted as ϵ_i for $i \in$

$[0, \tau]$, are then encrypted to level 1 ciphertexts, $[\![Z^{\alpha}_{IU}]\!]$ and $[\![\epsilon_{i,IU}]\!]$, with the querying SU's AFGH public key. Next, $[\![\mathbf{E}_{f,m,n,tp}]\!]$, $[\![\mathbf{S}_{f,m,n,tp}]\!]$, and $[\![\mathbf{O}_{f,m,n,tp}]\!]$ are re-encrypted from level 2 ciphertexts to level 1 ciphertexts with the SU's re-encryption keys. SAS then performs the following homomorphic calculation for each IU and corresponding Z^{α}_{IU}:

$$\forall i \in [0, \tau] : [\![Z^{\alpha*}_{i,IU}]\!] \leftarrow [([\![count_{SU,IU}]\!] \oplus [\![\epsilon_i]\!]^{-1}) \otimes R] \oplus [\![\mathbf{E}_{f,m,n,tp}]\!] \oplus [\![Z^{\alpha}_{IU}]\!] \quad (1)$$

in which R is a random, large, negative nonce. The first portion of this calculation, $([\![count_{SU,IU}]\!] \oplus [\![\epsilon_i]\!]^{-1}) \otimes R$, ensures that if the current value of $[\![count_{SU,IU}]\!]$ is greater than τ, the resulting decrypted $Z^{\alpha*}_{i,IU}$ will be equal to Z^{α}_{IU} distorted by a multiple of the random nonce, R. Additionally, if the query originated from a cell inside of the E-Zone, $[\![\mathbf{E}_{f,m,n,tp}]\!] \oplus [\![Z^{\alpha}_{IU}]\!]$ will distort the result by the random value in G_T to which $\mathbf{E}_{f,m,n,tp}$ was initialized. If the current value of $[\![count_{SU,IU}]\!]$ is less than τ and the query originated from a cell outside of the E-Zone, there will be one resulting $Z^{\alpha*}$ which is equal to Z^{α}_{IU}. SAS also produces one $[\![Z^{\alpha*}_{i,IU}]\!]$ for each IU based upon the obfuscated E-Zone map by simply calculating $[\![Z^{\alpha*}_{i,IU}]\!] \leftarrow [\![\mathbf{O}_{f,m,n,tp}]\!] \oplus [\![Z^{\alpha}_{IU}]\!]$. When decrypted, this $Z^{\alpha*}_{i,IU}$ will equal Z^{α}_{IU} if and only if the SU is located outside of the IU's obfuscated E-Zone and will allow the SU to recover a license if it is outside of the obfuscated E-Zone. Thus, $[\![Z^{\alpha*}_{i,IU}]\!]$ equals Z^{α}_{IU} when the request is from either of the valid SU request situations.

SAS then hashes each Z^{α}_{IU} with a different cryptographic hash function than was used as the primary hash function, which we shall refer to as the secondary hash function, and returns this list of hash digests, $[\![license]\!]_{AES,k,iv}$, and all resulting $[\![Z^{\alpha*}_{i,IU}]\!]$ rearranged in a randomized order, to the querying SU.

SU Operations. Each SU must initially register with SAS with the registration process specified by the FCC and may then query SAS by providing its current location, requested channel, and requested maximum transmit power. SAS will then respond to this query with a list of hash digests, $[\![license]\!]_{AES,k,iv}$, and a number of $[\![Z^{\alpha*}]\!]$ values. The SU then decrypts each $[\![Z^{\alpha*}]\!]$ using its private key and executes the secondary hash function on the resulting decrypted $Z^{\alpha*}$. If the result of this hash exists in the list of hash digests, the SU recognizes this digest corresponds to an IU's Z^{α}, recovers k and iv using the primary hash function, and removes it from the list. The SU then uses these k and iv values to remove one layer of encryption from $[\![license]\!]_{AES,k,iv}$. Once the $Z^{\alpha*}$ for each digest in the list has been found and the corresponding k and iv pair used to encrypt $[\![license]\!]_{AES,k,iv}$ has be recovered, the SU can successfully fully decrypt the license. If the SU hashes each $Z^{\alpha*}$ but is unable to find all digests in the list, its spectrum request was invalid and thus it is unable to recover the license.

Correctness of SZ-SAS. The correctness property requires that when an SU is located in an E-Zone of any IU, its spectrum request cannot be approved, and thus an SU cannot receive a valid spectrum license. Additionally, when an SU which has exceeded the query threshold of any IU and is located within the obfuscated E-Zone for this IU, its spectrum request cannot be approved. The SZ-SAS functionality can be donated as a function f:

$$\texttt{license}^* := f([\![\mathbf{E}_{f,m,n,tp}]\!], [\![\mathbf{O}_{f,m,n,tp}]\!], \tau, [\![count_{SU,IU}]\!], \texttt{req}), \qquad (2)$$

where req is the information received from an SU during a spectrum request.

Definition 1. *SZ-SAS is correct if it satisfies the following condition: For any input* $([\![\boldsymbol{E}_{f,m,n,tp}]\!], [\![\boldsymbol{O}_{f,m,n,tp}]\!], \tau, [\![count_{SU,IU}]\!], \texttt{req})$ *to SZ-SAS, if the requested location* (m, n) *is within an IU's E-Zone, or* $[\![count_{SU,IU}]\!] > \tau$ *and* (m, n) *is within an IU's obfuscated E-Zone,* license* *is invalid. Conversely, if the requested location* (m, n) *is outside of all IU's E-Zone, and* $[\![count_{SU,IU}]\!] < \tau$ *for all IUs or* (m, n) *is outside of all IU's obfuscated E-Zone,* license* *is valid.*

Theorem 1. *The probability with which SZ-SAS is NOT correct is negligible.*

Proof. The correctness follows directly from the specification of the SZ-SAS protocols. $[\![count_{SU,IU}]\!]$ can be updated using the homomorphic addition specified in Sect. 3.1. Let (m, n) be the location of req. If (m, n) is located within an IU's E-Zone, then $\mathbf{E}_{f,m,n,tp} \xleftarrow{\$} Z_p \backslash \{0\}$ for this IU. Thus, in the situation in which a query originates from the E-Zone of an IU, $Z_{IU}^{\alpha *} = Z_{IU}^{\alpha} + Z_p \backslash \{0\} \neq Z_{IU}^{\alpha}$. In the second situation, if $[\![count_{SU,IU}]\!] > \tau$, then $Z_{IU}^{\alpha *} = Z_{IU}^{\alpha} + R \neq Z_{IU}^{\alpha}$ where R is some random value in G_T. Additionally, if the query is also located in the obfuscated E-Zone of an IU, then $\mathbf{O}_{f,m,n,tp} \xleftarrow{\$} Z_p \backslash \{0\}$ for this IU. As a result, the $Z_{IU}^{\alpha *} = Z_{IU}^{\alpha} + Z_p \backslash \{0\} \neq Z_{IU}^{\alpha}$ for the $Z_{IU}^{\alpha *}$ associated with the obfuscated E-Zone map. As a result, for all i, $Z_{IU,i}^{\alpha *} \neq Z_{IU}^{\alpha}$, and as such, the key and IV for this IU will not be recoverable, and the license will not be successfully decrypted by the SU. Conversely, if (m, n) is located outside of an IU's E-Zone, then $\mathbf{E}_{f,m,n,tp} = 0$ for this IU and additionally if $[\![count_{SU,IU}]\!] < \tau$, then $Z_{IU}^{\alpha *} = Z_{IU}^{\alpha} + \mathbf{E}_{f,m,n,tp} = Z_{IU}^{\alpha}$. Also, if (m, n) is outside of an IU's obfuscated E-Zone map, then $\mathbf{O}_{f,m,n,tp} = 0$ for this IU. As a result, the $Z_{IU}^{\alpha *} = Z_{IU}^{\alpha}$ for the $Z_{IU}^{\alpha *}$ associated with the obfuscated E-Zone map and the SU can recover the license.

3 Preserving Location Privacy of Incumbent Users Against Inference Attacks

In this section, we describe the inference attack and analyze the compatibility and efficacy of potential obfuscation schemes.

3.1 Threat Model

We assume that there exists a singular honest-but-curious mobile SU with the ability to query SAS throughout the entirety of the region covered by SAS, this

SU shall henceforth be referred to as the adversary. The adversary's goal is to determine the grid location of a stationary IU using only information gained from the responses received from SAS. We assume the attacker has knowledge of $T(P_L, I_{th})$, the propagation model used by the IUs to calculate P_L, and I_{th}. We also assume the adversary has side knowledge indicating the existence of at least one IU operating on a channel of interest served by SAS.

3.2 Location Inference Algorithms

The Bayesian inference algorithm presented in [2] is robust and allows SUs to approximate the location of multiple IUs simultaneously. Thus, we will employ this algorithm to test the efficacy of our obfuscation techniques. However, we also propose a separate algorithm with the focus of locating a singular IU lacking the ability to inject false positive responses, as is the case of IUs in [3]. This algorithm emphasizes the importance of implementing protections against inference attacks and is introduced below.

First-Detected IU Inference Algorithm. We first define a Bernoulli random variable, $R_{xy}^{(k)}$, which represents the event of an IU existing in grid location $g(x, y)$ on channel k. Based upon the properties of the Bernoulli distribution, $P(R_{xy}^{(k)} = 1) = p_{xy}^{(k)}$ and $P(R_{xy}^{(k)} = 0) = 1 - p_{xy}^{(k)}$. Because the adversary has knowledge that there exists at least one IU on a channel of interest in the area covered by SAS, the IU is equally likely to be located in any of the grid locations covered by SAS. Thus, we initialize $p_{xy} = \frac{1}{MN}$ $\forall$ $g(x, y)$ on the given channel.

The adversary then queries the database from a chosen location and updates the value p_{xy} for all affected $g(x, y)$ based upon the response received from SAS. The adversary's inference regarding IU location for each possible combination of query responses is as follows:

- **Valid License for both TP_1 and TP_2:** This first case implies that there are no IUs operating in any cells with path loss values less than P_{L2} relative to the query location. The adversary first sets $p_{xy} = 0$ for all $P_{L_{xy}} < P_{L2}$, then adjusts p_{xy} for each remaining non-zero p_{xy} to reflect the current number of possible IU locations. As all non-zero locations are equally likely to contain the IU, $p_{xy} = \frac{1}{(MN)-n_{p0}}$, where n_{p0} is the total count of $g(x, y)$ with $p_{xy} = 0$.
- **Invalid License for TP_2, Valid License for TP_1:** In this case, the adversary's queries for TP_2 and TP_1 imply that there is an IU operating in some cell with $P_{L1} < P_{L_{xy}} < P_{L2}$ with respect to the queried location. The adversary uses this information to update the affected values of p_{xy} by first setting $p_{xy} = 0$ for all $P_{L_{xy}} > P_{L2}$ or $P_{L_{xy}} < P_{L1}$. The adversary then adjusts all remaining non-zero p_{xy} using the same logic as in the first scenario.
- **Invalid License for both TP_2 and TP_1:** This final case indicates the existence of an IU operating in a cell with $P_{L_{xy}} < P_{L_1} < P_{L_2}$. The adversary sets $p_{xy} = 0$ for all $P_{L_{xy}} > P_{L_1}$ and uses the same logic as in the first scenario to update all remaining non-zero p_{xy}.

The two inference attack algorithms can be compared by determining the number of queries required to locate the IU with some degree of certainty, which can be quantified either as the value of p_{xy} for the IU's actual location or as the calculated incorrectness (IC) as defined in [5], representing the distance between the actual and inferred location of the IU (Table 1).

Table 1. Queries required to geolocate IU using the two inference attack algorithms based upon two threshold metrics averaged over 1000 trials.

	IC < 0.01	p(x, y) > 0.90
Standard algorithm	142.12022	161.18045
Modified algorithm	115.75138	111.24599

3.3 Obfuscation Schemes

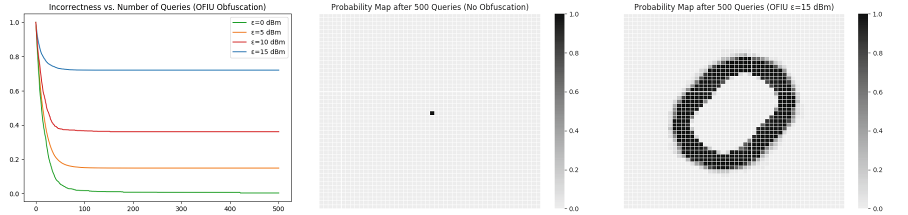
Previous works regarding obfuscation have focused entirely on techniques which could be applied directly by SAS. Because SZ-SAS encrypts IU parameters and SAS responses, obfuscation schemes cannot be applied by SAS and thus must be applied by the IUs. We propose a novel obfuscation scheme below.

Envelopment by Offset False IUs (OFIU). This obfuscation scheme covers the entirety of the true E-Zone map with exclusion zones generated by artificially generated IUs. In order to simulate each IU, we first select a random x and y location offset, and add this offset to the IU's true location to create a new location for the artificially generated IU. We then select a random negative noise value for each, and add this to the actual IU's maximum transmit power level. We adjust these parameters until the true IU's exclusion zone is completely enveloped by the newly generated IU exclusion zones. This results in a new exclusion map which is not centered around the IU's true location.

3.4 Experimental Results

We conducted a series of experiments testing each of the proposed obfuscation schemes. We consider an area covered by SAS to be a 50 × 50 grid with grid side lengths of 250 m, a total area of 156.25 km^2. In this situation, the total communication overhead for each SU request is only 3 kB and 231 kB for SAS responses when the region contains 100 IUs.

Figure 2a shows that as the zone is enlarged by OFIU, the lower bound of incorrectness increases, implying that an adversary should not be successful in locating the IU. Additionally, the adversary is able to accurately locate the IU when obfuscation is not applied as in 2b, but is unable to locate this same IU when obfuscation is applied in 2c.



(a) Incorrectness at various levels of OFIU obfuscation

(b) Result of SU inference attack without obfuscation

(c) Result of SU inference attack with OFIU

Fig. 2. Results of SU inference attack on IUs with and without OFIU obfuscation

4 Conclusion

A novel framework which protects sensitive IU parameters from both untrusted SUs and untrusted SAS was successfully developed. We demonstrate the effectiveness of an inference attack, and show the necessity of obfuscation in the prevention of such an attack. Our experimental analysis demonstrates the ability of our framework to utilize compatible obfuscation schemes to prevent such an inference attack. In future works, this framework can be utilized as a building block when designing SAS-based DSA systems. Other obfuscation methods may also be developed and analyzed. Additionally, as light-weight fully homomorphic cryptosystems are developed, AFGH may be replaced by one such cryptosystem in our framework, which would allow researchers more flexibility when designing novel obfuscation schemes.

References

1. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. ACM Trans. Inf. Syst. Secur. **9**(1), 1–30 (2006). https://doi.org/10.1145/1127345.1127346
2. Bahrak, B., Bhattarai, S., Ullah, A., Park, J.M.J., Reed, J., Gurney, D.: Protecting the primary users' operational privacy in spectrum sharing. In: 2014 IEEE International Symposium on Dynamic Spectrum Access Networks (DYSPAN), pp. 236–247, April 2014. https://doi.org/10.1109/DySPAN.2014.6817800
3. Dou, Y., et al.: Preserving incumbent users' privacy in server-driven dynamic spectrum access systems. In: 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS), pp. 729–730, June 2016. https://doi.org/10.1109/ICDCS.2016.40
4. Park, J.M., Reed, J.H., Beex, A.A., Clancy, T.C., Kumar, V., Bahrak, B.: Security and enforcement in spectrum sharing. Proc. IEEE **102**(3), 270–281 (2014). https://doi.org/10.1109/JPROC.2014.2301972

5. Shokri, R., Theodorakopoulos, G., Le Boudec, J.Y., Hubaux, J.P.: Quantifying location privacy. In: Proceedings of the 2011 IEEE Symposium on Security and Privacy, SP 2011, pp. 247–262. IEEE Computer Society, Washington, DC (2011). https://doi.org/10.1109/SP.2011.18
6. Zhang, L., Fang, C., Li, Y., Zhu, H., Dong, M.: Optimal strategies for defending location inference attack in database-driven CRNs. In: 2015 IEEE International Conference on Communications (ICC), pp. 7640–7645, June 2015. https://doi.org/10.1109/ICC.2015.7249548