Journal of Computer Science and Frontier Technologies
ISSN Print: 3104-4204
ISSN Online: 3104-4212
CD ISSN: 3104-4220
https://doi.org/10.63313/JCSFT.9048

GP-STCKS: A GeoPrefix-based Spatio-Temporal Controllable Keyword Search Scheme for Privacy-Preserving Electronic Health Systems

ZiXi Dong
Qingdao University, Qingdao 266071, China
Email: dongzixi@qdu.edu.cn

How to cite this paper: Dong, Z. X. (2026). GP-STCKS: A GeoPrefix-based Spatio-Temporal Controllable Keyword Search Scheme for Privacy-Preserving Electronic Health Systems. *Journal of Computer Science and Frontier Technologies*, 2(3), 61-70. ISSN Print: 3104-4204; ISSN Online: 3104-4212.
https://doi.org/10.63313/JCSFT.9048
Published: 2026-03-09

Copyright © 2026 by author(s) and Erytis Publishing Limited.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/



Abstract

The proliferation of cloud-based e-health systems necessitates outsourcing sensitive Electronic Health Records (EHRs), raising critical privacy and access control concerns. While Attribute-Based Keyword Search (ABKS) enables secure encrypted retrieval, existing schemes inherently lack fine-grained spatio-temporal constraints. To address this, we propose GP-STCKS, a novel privacy-preserving spatio-temporal controllable keyword search scheme. By seamlessly integrating Geohash hierarchical indexing into the Ciphertext-Policy ABKS (CP-ABKS) framework, our scheme enforces strict dual access constraints over specific time intervals and spatial hierarchies. Rigorous formal analysis proves its resilience against chosen-keyword and multi-dimensional key collusion attacks while guaranteeing keyword confidentiality. Extensive experiments demonstrate that GP-STCKS maintains lightweight overhead and optimal search efficiency, perfectly balancing stringent privacy protection with real-world e-health deployability.

Keywords

Attribute-Based Keyword Search; Spatio-Temporal Access Control; Location Privacy; Geohash Indexing; E-health Systems

1. Introduction

E-health systems increasingly rely on cloud infrastructure for the efficient management and sharing of medical data. To protect data privacy against untrusted cloud providers, records are encrypted before outsourcing, driving the adoption of Attribute-Based Keyword Search (ABKS)[1][2] to enable secure and fine-grained data retrieval. However, existing ABKS schemes lack strict temporal and spatial constraints[3]. In practical e-health scenarios, this limitation allows authorized users to execute queries outside of permitted times or locations. This vulnerability introduces two major risks: (1) Unauthorized Access: Querying data beyond

temporal or spatial permissions can lead to untraceable data breaches. (2) Privacy Leakage: Attackers or semi-honest cloud servers can analyze query patterns to infer user routines, locations, and sensitive patient associations.

To address the lack of spatio-temporal constraints in traditional cloud medical data retrieval, we utilized Geohash hierarchical indexing, a multi-dimensional time derivation function, and the CP-ABKS framework to construct a GeoPrefix-based spatio-temporal controllable keyword search scheme (GP-STCKS) equipped with HMAC-based location matching identifiers.

2. Preliminary

2.1. Location Hierarchy Matching Function

We represent geographical locations using Geohash encoding, which maps latitude–longitude coordinates into hierarchical prefixes. Let $G = \{G_1, G_2, \dots, G_m\}$ denote the prefix set of a location, where G_i is the prefix at level i.

To support privacy-preserving hierarchical location verification, we define a pseudorandom function $F_{k_{geo}}(\cdot)$ with secret key k_{geo} generated by the trusted authority. Using an HMAC construction, it maps each prefix to a secure location identifier by (1):

$$\delta_{G_i} = F_{k_{geo}}(G_i, level_i) = HMAC_{k_{geo}}(G_i \parallel level_i). \quad (1)$$

The function provides irreversibility and hierarchy distinguishability, ensuring location privacy while enabling secure multi-level prefix matching.

3. Our Construction OF GP-STCKS

3.1. System Model

As illustrated in **Figure 1**, the proposed GP-STCKS framework comprises five primary entities:

- **Trusted Authority (TA):** A fully trusted entity that initializes the system, distributes secret keys, and computes location matching identifiers.
- **Data Owner (DO):** Encrypts electronic health records (EHRs) and generates keyword indexes. The DO outsources encrypted data, attribute indexes, and spatial indexes to the CS, while separately sending time indexes to the TS.
- **Data Users (DUs):** Authorized personnel who generate search tokens to query the encrypted data based on their attributes.
- **Cloud Server (CS):** Stores ciphertexts and indexes, verifies location matching identifiers, and executes search operations using the tokens submitted by DUs.
- **Time Server (TS):** Securely stores and transforms the time indexes.

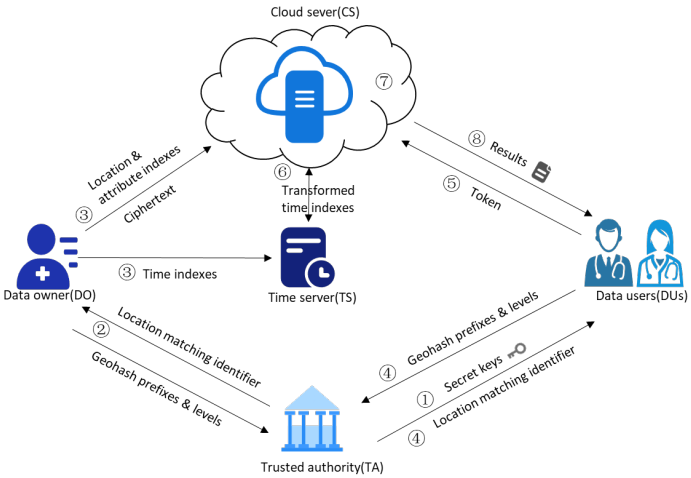


Figure 1. System model of our GP-STCKS.

3.2. Framework Construction

The proposed framework consists of five algorithms, namely Setup, KeyGen, Enc, Token, and Search, which are described as follows.

Setup(1^κ) : Given a system security parameter κ, TA generates the composite-order bilinear group parameters $\Omega = (N, \mathbb{G}, \mathbb{G}_T, e)$[4] and selects generators $g_0 \in \mathbb{G}, g_1 \in \mathbb{G}_{s'}, \varphi \in \mathbb{G}_{n'}$. Next, TA randomly chooses elements $a, b, c \in \mathbb{Z}_n^*$ and $\lambda, \mu \in \mathbb{Z}_n^*$. It also selects a cryptographic key $k_{geo} \xleftarrow{R} \{0,1\}^\kappa$ to serve as the secret key for the location hierarchy matching function $F_{k_{geo}}(\cdot)$.Subsequently, TA specifies four collision-resistant hash functions: $H_1: \{0,1\}^* \to \mathbb{G}_s$, $H_2: \{0,1\}^* \to \mathbb{Z}_n^*$, $H_3: \{0,1\}^* \to \mathbb{G}_s$, $H_4: \{0,1\}^* \to \mathbb{Z}_n^*$ .In addition, TA defines a multi-dimensional range derivation function $F_{time}(\cdot)$[5], which is securely shared between TS and authorized DUs. Finally, TA publishes the public key pk and securely retains the master secret key msk by (2):

$$pk = (\Omega, \lambda, \mu, g_0, g_1, \varphi, g_0^a, g_0^b, g_1^c, H_1, H_2, H_3, H_4),$$
$$msk = (p, q, n', a, b, c, k_{geo}). \quad (2)$$

KeyGen(msk, pk, S): Taking msk, pk and a user attribute set S as input, TA generates the private key sk. The attribute set is defined as $S = (S_A, time, geo)$, comprising descriptive attributes $S_A = \{at_j\}$, an authorized time interval $time = [t_s', t_e']$, and an authorized geographic region denoted by a Geohash prefix set $G' = \{G_1', G_2', \ldots, G_{m_1}'\}$.

First, TA selects $r \in \mathbb{Z}_n^*$ and computes $D = g_1^{(ac-r)/b}$.Then, for each $at_j \in S_A$, it selects $r_j \in \mathbb{Z}_n^*$ and computes: $D_j = g_1^r H_1(at_j)^{r_j}, D_j' = g_0^{r_j}$. For the time attribute $time$, it selects $r_{time} \in \mathbb{Z}_n^*$ and computes: $D_{time} = g_1^r H_1(time)^{r_{time}}, D_{time}' = g_0^{r_{time}}, D_{time}'' = v_{[t_s', t_e']}^{r_{time}}$. For each $G_i' \in G'$, it selects $r_{geo,i} \in \mathbb{Z}_n^*$ and computes the key components along with the corresponding location matching identifiers using $F_{k_{geo}}(\cdot): D_{geo,i} = g_1^r H_3(G_i')^{r_{geo,i}}, D_{geo,i}' = g_0^{r_{geo,i}}, \delta_{G_i'} = F_{k_{geo}}(G_i', level_i)$. Finally, TA outputs the generated private key sk for DU by (3):

$$sk = \left(S, D, \{(D_j, D_j')\}_{at_j \in S_A}, D_{time}, D_{time}', D_{time}'', \left\{\left(D_{geo,i}, D_{geo,i}', \delta_{G_i'}\right) | 1 \le i \le m_1\right\}\right). \quad (3)$$

$Enc(pk, R, W)$: Taking pk, the electronic health record set $R = \{R_i\}$, and the

extracted keyword set $W = \{w\}$ as input, DO first encrypts each record R_i using a symmetric encryption algorithm. Then it constructs the indexes for W.

DO constructs an access tree $T[6]$ over the descriptive attribute set S_A. For each node x in T, DO defines a polynomial $P_x(\cdot)$ of degree $d_x = k_x - 1$. It samples a secret $s \in \mathbb{Z}_n^*$ setting $P_{root}(0) = s$ for the root node, and recursively sets $P_x(0) = P_{parent(x)}\big(index(x)\big)$ for child nodes.

For each keyword $w \in W$, the index components are generated as follows:

• Descriptive attribute: For each leaf node l in the leaf set L of T, DO computes: $C_l = g_0^{P_l(0)}, C_l' = H_1\big(att(l)\big)^{P_l(0)}$.

• Time attribute: Given the data validity period $[t_s, t_e]$, DO selects $t_{time} \in \mathbb{Z}_n^*$ and computes: $C_{time} = g_0^{t_{time}}, C'_{time} = H_1(time)^{t_{time}}, C''_{time} = v_{[t_s,t_e]}^{t_{time}}$.

• Location attribute: DO selects $\rho \in \mathbb{Z}_n^*$ and computes $C = (g_1^c)^\rho$. For the valid geographic region defined by the Geohash prefix set $G = \{G_1, G_2, \dots, G_{m_2}\}$, DO sends the specified Geohash prefix and levels information $(G_i \parallel level_i)$ to TA.TA returns the corresponding location matching identifiers $\{\delta_{G_1}, \delta_{G_2}, \dots, \delta_{G_{m_2}}\}$. Upon receiving them, for each $G_i \in G$, the DO computes: $t_{geo,i} = H_4(G_i), C_i' = (g_0^b)^{s+t_{time}+t_{geo,i}}, C_{0,i} = (g_0^a)^{s+\rho+t_{time}+t_{geo,i}} \cdot (g_0^b)^{H_2(w)\rho}, C_{geo,i} = g_0^{t_{geo,i}}, C'_{geo,i} = H_3(G_i)^{t_{geo,i}}$.

Finally, the composite index is defined as $cph = \big(T, \{I_w, I_{time}, I_{geo}\}\big)$, where: $I_w = (C, \{(C_l, C_l')\}_{l\in L}), I_{\text{time}} = (C_{\text{time}}, C'_{\text{time}}, C''_{\text{time}}), I_{\text{geo}} = \{(C_{geo,i}, C'_{geo,i}, C_i', C_{0,i}, \delta_{G_i}) | 1 \le i \le m_2\}$.The DO sends the ciphertexts, T, I_w, and I_{geo} to the CS, while securely transmitting I_{time} to the TS.

$Token(w', sk, pk)$: When a DU queries the keyword w' at time t_c from a geographic location with the Geohash code G_U, the DU selects a random value $u \in \mathbb{Z}_n^*$ to generate the query token TK. The detailed procedure is as follows.

First, DU computes: $tok_1 = \left(g_0^a g_0^{bH_2(w')}\right)^u, tok_2 = g_1^{cu}, tok_3 = D^u = g_1^{(ac-r)u/b}$.

For each $at_j \in S_A$, DU computes: $\big(D_j\big)^u = g_1^{ur} H_1\big(at_j\big)^{ur_j}, \big(D_j'\big)^u = g_0^{ur_j}$.For the time attribute, DU components $(D_{time})^u = g_1^{ur} H_1(time)^{ur_{time}}, (D'_{time})^u = g_0^{ur_{time}}, (D''_{time})^u = F_{time}\left(v_{[t_s',t_e']}^{ur_{time}}\right) = v_{[t_c,t_c]}^{ur_{time}}$.: For the location attribute, given the current query location $G_U = \{G_{U,1}, G_{U,2}, \dots, G_{U,m_3}\}$, DU sends the specified Geohash prefix and levels information $\big(G_{U,k} \parallel level_k\big)$ to TA.TA computes and returns the location matching identifiers $\{\delta_{G_{U,1}}, \delta_{G_{U,2}}, \dots, \delta_{G_{U,m_3}}\}$ .Next, DU extracts the authorized location prefix set G' and its depth m_1 from sk. For each authorized prefix $G_i' \in G'$, the DU computes: $\big(D_{geo,i}\big)^u = g_1^{ur} H_3(G_i')^{ur_{geo,i}}, \big(D'_{geo,i}\big)^u = g_0^{ur_{geo,i}}$.

Finally, the DU assembles the location matching identifiers $\{\delta_{G_i'}\}$ contained in sk together with the above components to construct the query token TK. The DU submits TK and its attribute set S to the CS for search execution. The query token TK is defined as (4):

$$TK = (tok_1, tok_2, tok_3, \{\big((D_j)^u, (D_j')^u\big)\}_{at_j \in S_A}, (D_{time})^u, (D'_{time})^u, (D''_{time})^u, \left\{\left((D_{geo,i})^u, (D'_{geo,i})^u, \delta_{G_i'}\right) | 1 \le i \le m_1\right\}, \{\delta_{G_{U,k}} | 1 \le k \le m_3\}). \quad (4)$$

$Search(TK, pk, cph)$: After receiving the query token TK and the cph from DU, CS executes the matching verification algorithm.

First, CS interacts with TS to securely transform the queried time interval with respect to the current time t_c, retrieving the transformed component $C_{time}^* =$

$C_{time} \cdot F_{time}(C''_{time}) = \left(g_0 \cdot v_{[t_c,t_c]}\right)^{t_{time}}$. Subsequently, CS performs matching operations across three distinct dimensions:

- Descriptive attribute: CS evaluates if the DU's attribute set S_A satisfies the access tree T. If valid, it selects a satisfying subset $S'_A \subseteq S_A$ and recursively reconstructs the root value via polynomial interpolation to yield: $E_{root} = e(g_1, g_0)^{rus}$.
- Time attribute: If $t_c \in [t_s, t_e] \wedge t_c \in [t'_s, t'_e]$, CS computes by (5):

$$E_{\text{time}} = \frac{e\left((D_{\text{time}})^u, C^*_{\text{time}}\right)}{e\left((D'_{\text{time}})^u, C'_{\text{time}}\right)} = e(g_1, g_0)^{rut_{time}}. \tag{5}$$

- Location attribute: CS performs the following double location matching verification:$\delta_{G_{U,m_1}} = \delta_{G'_{m_1}} \wedge \delta_{G_{U,m_2}} = \delta_{G_{m_2}}$.If the above condition does not hold, the search procedure is aborted. Otherwise, the verification succeeds. The CS then determines the matching level according to the minimum common level strategy, where $L = min(m_1, m_2)$. The corresponding value $E_{geo,L}$ is computed by (6):

$$E_{geo,L} = \frac{e\left((D_{geo,L})^u, C_{geo,L}\right)}{e\left((D'_{geo,L})^u, C'_{geo,L}\right)} = e(g_1, g_0)^{rut_{geo,L}}. \tag{6}$$

Finally, the CS checks whether the query token TK matches the keyword index of w by (7):

$$e\left(C_{0,L}, tok_2\right) = e(C, tok_1) \cdot e(C'_L, tok_3) \cdot E_{root} \cdot E_{time} \cdot E_{geo,L}. \tag{7}$$

If the equation holds exactly, the CS returns the corresponding search results to the DU; otherwise, it returns 0.

4. Security Analysis

In this section, the ensuing security proofs rely on the generic bilinear group (GBG) model security of the underlying TCKS framework and the computational hardness of the RSA and co-CDH assumptions.

Theorem 1. Under the generic bilinear group (GBG) model and assuming H_4 is a random oracle, the proposed GP-STCKS scheme is semantically secure against CKA, provided the underlying TCKS framework is secure.

Proof. To break the CKA security of our scheme, a PPT adversary $\mathcal{A}$ must distinguish a valid keyword index by eliminating the composite blinding exponent $r\left(s + t_{time} + t_{geo,i}\right)$. As established in the TCKS security proof, canceling the base term $r(s + t_{time})$ is computationally infeasible without possessing valid attribute private keys that satisfy the target access policy. Furthermore, eliminating the newly introduced spatial term $r \cdot t_{geo,i}$ strictly requires the legitimate location private key component $D_{geo,i}$ to compute the necessary bilinear pairing $e(g_1, g_0)^{rut_{geo,i}}$. Since the random secret r is uniformly sampled and securely hidden, $\mathcal{A}$ cannot isolate this factor without valid location credentials. Therefore, successfully breaking GP-STCKS inherently requires breaking the underlying TCKS scheme. Consequently, the adversary's advantage is strictly negligible. □

Theorem 2. Under the co-CDH and RSA[7] assumptions, the proposed GP-STCKS scheme effectively resists multi-dimensional key collusion attacks across descriptive, time, and location attributes.

Proof. Resistance is rigorously guaranteed via strict algebraic non-separability and

root-key binding mechanisms. First, within single dimensions, extracting user-specific randomness from descriptive, time, or location key components (e.g., $D_{geo,i}$) is computationally equivalent to solving the co-CDH problem, while temporal interval expansion is thwarted by the RSA assumption.

Second, against cross-dimensional collusion (e.g., pooling valid components from distinct users U_A and U_B), the TA fundamentally binds each user's components to a globally unique random value ($r_A \neq r_B \in \mathbb{Z}_n^*$). Constructing a forged hybrid token TK^* evaluates to a mismatched pairing exponent $u(r_A s + r_A t_{time} + r_B t_{geo,L})$ during verification. Under the discrete logarithm assumption, colluding users cannot algebraically extract a unified common factor r to cancel the blinding terms and satisfy the verification equation. Thus, collusion is mathematically impossible, and the adversary's advantage is negligible.

5. Performance Analysis

5.1. Theoretical Complexities

We theoretically compare the proposed GP-STCKS scheme with the underlying TCKS scheme[3], which solely supports time-controlled search. The analysis encompasses four core phases: key generation, encryption, token generation, and search. To quantify the overhead effectively, we define the primary cryptographic operations and parameters as follows:

Table 1. Theoretical Efficiency Analysis and Comparison

		GP-STCKS	TCKS
Computation costs	Keygen	$(5+2m_1)E + (1+m_1)H + (2E+H)\|M\|$	$5E + H + (2E+H)\|M\|$
	Enc	$H + (4+\|W\|)E + (2E+H)\|V\| + (4E+H)m_2$	$H + (6+\|W\|)E + (2E+H)\|V\|$
	Token	$5E + qE + 2\|M\|E + 2m_1E$	$5E + qE + 2\|M\|E$
	Search	$(\|W\|+q+5)P + 2\|M\|P$	$(\|W\|+q+3)P + 2\|M\|P$
Storage costs	Keygen	$4\|G\| + 2\|G\|\|M\| + 2m_1\|G\| + m_1\|\delta\|$	$4\|G\| + 2\|M\|\|G\|$
	Enc	$(2\|V\|+4)\|G\| + m_2(3\|G\| + \|W\|\|G\| + \|\delta\|)$	$(5+\|W\|+2\|V\|)\|G\|$
	Token	$5\|G\| + q\|G\| + 2\|M\|\|G\| + m_1(2\|G\| + \|\delta\|) + m_3\|\delta\|$	$5\|G\| + q\|G\| + 2\|M\|\|G\|$
	Search	$(\|W\|+q+4)\|G_T\| + \|M\|\|G_T\|$	$(\|W\|+q+3)\|G_T\| + \|M\|\|G_T\|$

M:number of descriptive attributes; V:number of leaf nodes in the access policy; $|W|$:number of keywords in the dictionary; q: number of queried keywords; $|\delta|$:output length of the location identifier.

E denotes a modular exponentiation, P represents a bilinear pairing, and H denotes a hash mapping to the subgroup $\mathbb{G}_{s'}$ (including H_1 and H_3). Highly efficient operations (e.g., $H_2, H_4 \to \mathbb{Z}_n^*$ and the HMAC-based $F_{k_{geo}}$) incur strictly negligible overhead and are omitted. $|\mathbb{G}|$ and $|\mathbb{G}_T|$ denote the bit-lengths of elements in the groups $\mathbb{G}$ and $\mathbb{G}_T$ respectively. The parameters m_1, m_2, m_3 represent the number of hierarchical levels of G', G, G_U. **Table 1** presents a comprehensive theoretical efficiency comparison between the proposed GP-STCKS scheme and the TCKS scheme in terms of computational cost and storage overhead.

5.2. Experimental results

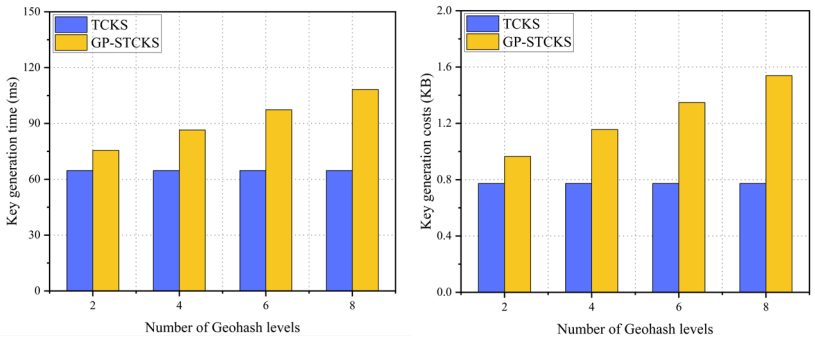
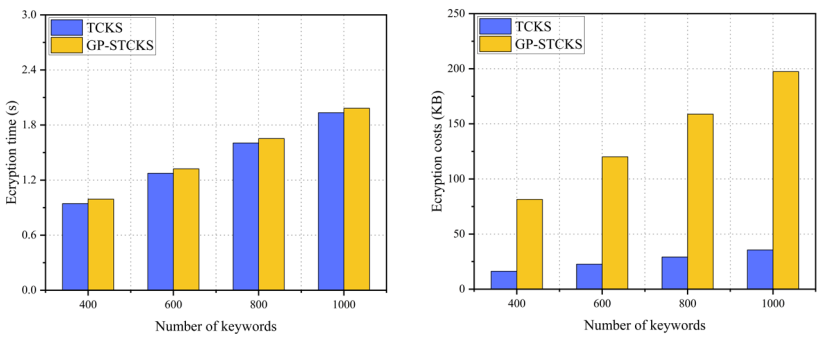
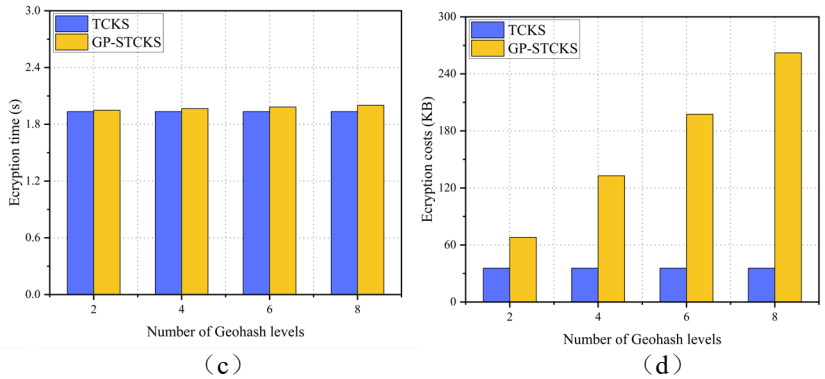


Figure 2. Performance of $KeyGen$.

Figure 2 illustrates the $KeyGen$ overhead with $|M| = 10$ and $m_1 \in [2,8]$. Unlike the TCKS baseline, which maintains constant overhead due to its lack of spatial constraints, both the computation and storage costs of GP-STCKS grow linearly with m_1. This linear growth stems from generating independent key components $(D_{geo,i}, D'_{geo,i})$ and location matching identifiers $\delta_{G'_i}$ for each authorized hierarchy level.Nevertheless, even at the maximum spatial precision ($m_1 = 8$), the total execution time and key size remain strictly at the millisecond and kilobyte levels. This clearly demonstrates that GP-STCKS successfully achieves fine-grained spatial access control with negligible overhead, ensuring excellent lightweight efficiency for practical system deployments.

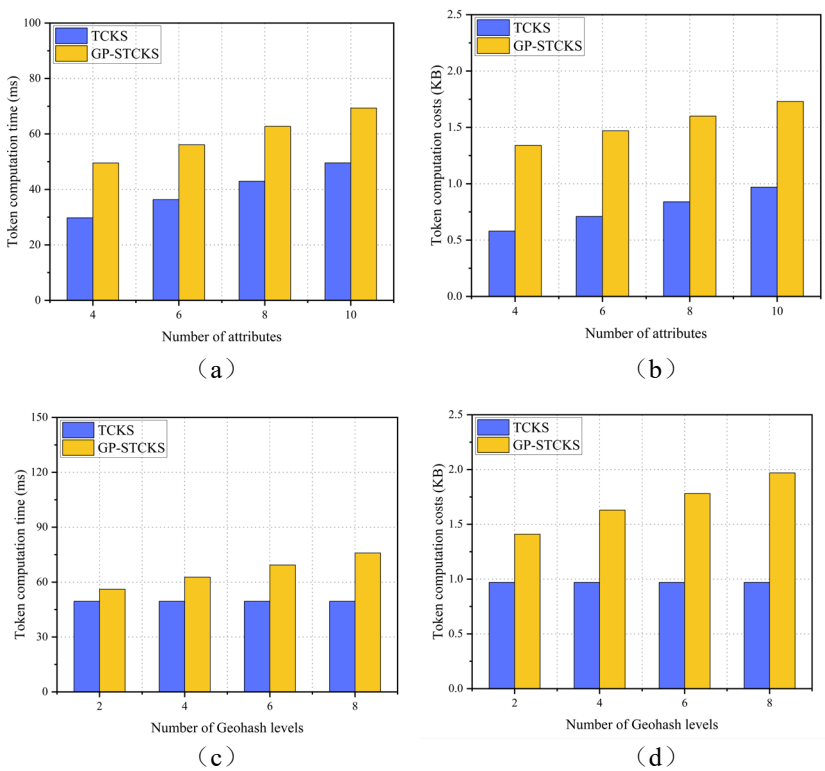




（c） （d）

Figure 3. Performance of *Encryption*.

Figure 3 evaluates the *encryption* overhead against the keyword set size $|W|$ and the spatial hierarchy depth m_2 (with $|V| = 50, |M| = 10$). While GP-STCKS maintains computational efficiency nearly identical to the TCKS baseline, its storage overhead grows linearly at approximately m_2 times the rate of TCKS. Because the system needs to compute and store independent index components for each location hierarchy. Consequently, the overall overhead scales linearly with m_2. However, since index construction is inherently a one-time operation executed by the data owner, the incremental latency and storage overhead remain highly practical for real-world deployments, even under maximum parameters ($m_2 = 8, |W| = 1000$).



（a） （b）

（c） （d）

Figure 4. Performance of $Token$.

Figure 4 evaluates the token generation overhead against the user attribute count $|M|$ and the authorized spatial hierarchy depth m_1. When varying $|M|$, GP-STCKS and the baseline TCKS exhibit identical linear growth rates, with our scheme introducing merely a fixed, constant-cost increment for spatial token generation. Conversely, when varying m_1, the overhead of TCKS remains static, whereas GP-STCKS scales slightly and linearly to compute the required spatial components and location matching identifiers. Crucially, even at the maximum spatial control precision, the additional latency introduced by GP-STCKS remains strictly at the millisecond scale. This confirms that our scheme seamlessly achieves fine-grained physical spatial access control with imperceptible delays, rendering it highly practical for deployment on resource-constrained mobile devices.

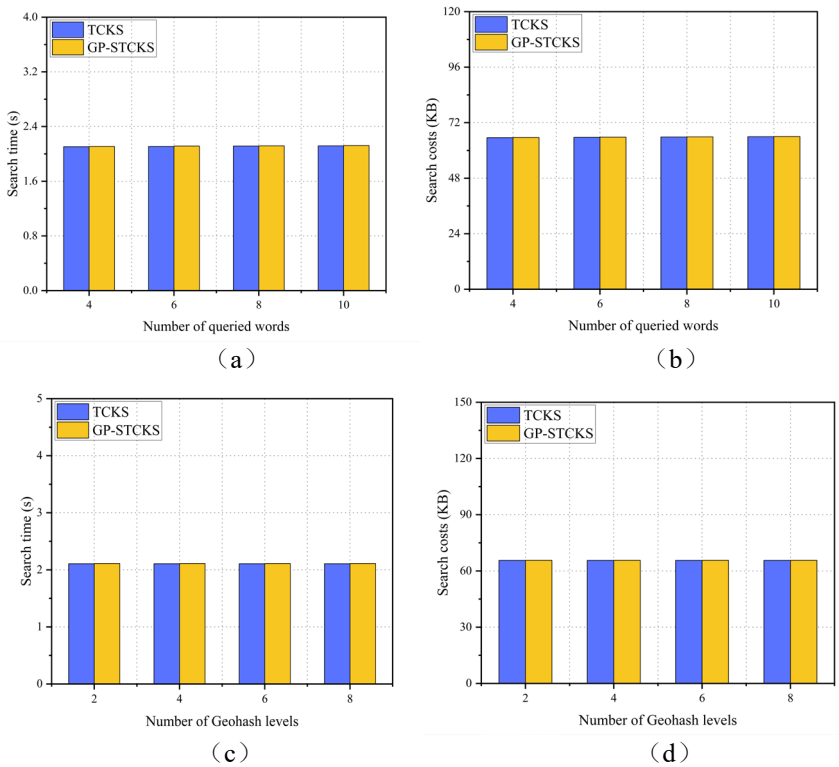


Figure 5. Performance of $Search$.

Figure 5 evaluates the search phase performance against the queried keyword count q and the location matching hierarchy level L. Since practical queries satisfy $q \ll |W|$, varying q exerts a strictly negligible impact on the system response time. Crucially, the search overhead remains fundamentally constant across any spatial hierarchy L. Because index components for different location levels are independently precomputed during the encryption phase, the cloud server can directly execute exact or downgrade matching operations with optimal $O(1)$ complexity. This optimal design ensures that GP-STCKS rigorously enforces

fine-grained spatial control while seamlessly satisfying the stringent real-time demands of large-scale cloud services.

6. Conclusion

This paper proposes GP-STCKS, a privacy-preserving spatio-temporal controllable keyword search scheme. By integrating Geohash hierarchical indexing with CP-ABKS and HMAC-based location matching identifiers, the scheme achieves precise spatial access control and supports a highly flexible downgrade matching strategy. Formal analysis proves its robust security against chosen-keyword and multi-dimensional key collusion attacks, while fully preserving keyword confidentiality. Extensive experiments demonstrate that GP-STCKS incurs lightweight overhead and delivers fast, stable retrieval. Ultimately, our scheme successfully achieves an optimal balance between stringent location privacy protection and the practical efficiency demands of real-time cloud services.

References

[1] Q. Zheng, S. Xu, and G. Ateniese, "VABKS: Verifiable attribute-based keyword search over outsourced encrypted data," in Proc. IEEE Conf. Comput. Commun., 2014, pp. 522–530.

[2] W. Sun, S. Yu, W. Lou, Y. T. Hou, and H. Li, "Protecting your right: Attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud," in Proc. IEEE Conf. Comput. Commun., 2014, pp. 226–234.

[3] Y. Miao et al., "Time-Controllable Keyword Search Scheme With Efficient Revocation in Mobile E-Health Cloud," in IEEE Transactions on Mobile Computing, vol. 23, no. 5, pp. 3650-3665, May 2024, doi: 10.1109/TMC.2023.3277702.

[4] Zhu Y, Hu H, Ahn G J, et al. Comparison-based encryption for fine-grained access control in clouds[C]//Proceedings of the second ACM conference on Data and Application Security and Privacy. 2012: 105-116.

[5] Z. Wang, D. Huang, Y. Zhu, B. Li, and C.-J. Chung, "Efficient attributebased comparable data access control," IEEE Trans. Comput., vol. 64, no. 12, pp. 3430–3443, Dec. 2015.

[6] Bethencourt J, Sahai A, Waters B. Ciphertext-policy attributebased encryption. In: Security and Privacy, 2007. SP'07. IEEE symposium on. IEEE; 2007. p. 321–34.

[7] S. Hohenberger and B. Waters, "Synchronized aggregate signatures from the RSA assumption," in Proc. Annu. Int. Conf. Theory Appl. Cryptogr. Techn., Springer, 2018, pp. 197–229.