A Security Analysis

The security analysis of our query rewriter is based on the composition lemma in [27] (see its Sec. 7.3.1). The lemma states that given a secure protocol $\pi^{g|f}$ that can securely compute g based on a plaintext query f and a secure protocol π^f for operation f, the operation g can be securely computed by executing protocol $\pi^{g|f}$ but substitutes every plaintext query f with an execution of π^f.

We prove the security of query rewriter for asymmetric federated queries (Appendix A.1) and symmetric federated queries (Appendix A.2) separately. Finally, we present a case study on attacking asymmetric federated range counting in Appendix A.3.

A.1 Security Analysis of Asymmetric Federated Queries

In the following, we conduct a thorough analysis of the security of asymmetric federated queries in Hu-Fu, categorizing them into two distinct kinds: *radius-known queries* and *radius-unknown queries*.

- **Security of Radius-Known Queries.** Federated range query, range counting and distance join belong to *radius-known queries*. For *federated range query* and *distance join*, the secure protocol $\pi^{g|f}$ is the secure set union (Def. 5) based on the query results of one or multiple plaintext range queries over the federation. The protocol π^f corresponds to the plaintext range query (Def. 1). The protocol π^f is secure, since the plaintext range query only occurs in each silo, which will not leak any information to other silos. From the composition lemma, federated range query and distance join are secure against semi-honest adversaries. Federated range counting is also secure based on a similar proof (*i.e.*, replacing $\pi^{g|f}$ with the secure summation in Def. 3 and π^f with the plaintext range counting in Def. 1).
- **Security of Radius-Unknown Queries.** The federated kNN query and kNN join belong to *radius-unknown queries*. For federated kNN query (Alg. 1), let protocol π^f be the secure count comparison between k and the local counts. The security of π^f can be proved in the same way as before. Denote protocol $\pi^{g|f}$ as Alg. 1 based on the plaintext count comparison query. We then show protocol $\pi^{g|f}$ is also secure. Assuming semi-honest adversaries, each silo can simulate its view of the execution of the protocol. Specifically, knowing the range $[l, u]$ used at the beginning of a round, each silo can compute r used in that round. If r is the same as the final radius, it concludes that the protocol must have ended in this round. Otherwise, it simply updates the range to that side of r which contains the final radius. Along with the knowledge of the initial range, this shows that each silo can simulate the execution of the protocol, which means the protocol $\pi^{g|f}$ is secure. Thus, our Alg. 1 is secure against semi-honest adversaries based on the composition lemma. The security of federated kNN join can be proved similarly.

A.2 Security Analysis of Symmetric Federated Queries

Similar to the previous security analysis, we analyze the security of symmetric federated queires in Hu-Fu from two categories: *radius-known queries* and *radius-unknown queries*.

Table 5: A case study on hacking (asymmetric) federated range counting in Hu-Fu.

Colluded silos	Attack success rate	Running time
1	0	No result for 2 months
2	0	No result for 2 months
3	0	No result for 2 months

- **Security of Radius-Known Queries.** For *federated range query*, the secure protocol $\pi^{g|f}$ employs the secure distance comparison (Def. 6), which relies on the fully homomorphic encryption (FHE) scheme (*i.e.*, the BGV scheme [11]), and the secure set union (Def. 5). The protocol π^f corresponds to the plaintext range query (Def. 1). It is secure because the plaintext range query is executed within each silo individually, thereby preventing any information leakage to other silos. According to the composition lemma, the query rewriter for symmetric federated range queries ensures data privacy. Regarding the additional privacy requirement of query privacy, the secure protocol, secure location perturbation (Def. 7), is based on the BPL mechanism in [61]. This mechanism has been proven to satisfy (ϵ, δ)-Geo-Indistinguishability (Geo-I) [12], a widely-adopted privacy notion for preserving location privacy. Thus, the query privacy is due to the protection of query location with secure location perturbation. As for *federated distance join*, the data privacy and query privacy can be guaranteed, since a federated distance join is decomposed into a series of federated range queries, and the security of each federated range query has been proven. The security of *federated range counting* shares similarities with that of federated range query, primarily due to their near-identical decomposition plans (see Table 3). However, a key difference lies in the nature of their query answers. While federated range query requires a secure set union to collect spatial objects falling within the specified range, federated range counting simply returns an integer count as its output. This distinction underscores the minimal exposure of information in federated range counting, further reinforcing its security.
- **Security of Radius-Unknown Queries.** To prove the security of radius-unknown queries, we only need to show the security of federated kNN query, since a federated kNN join is decomposed into multiple independent federated kNN queries by Hu-Fu. Similar to the decomposition plan for asymmetric federated kNN query, the query rewriter for symmetric federated kNN query also utilizes a binary-search procedure. The key difference is that the initial upper bound is determined by plaintext kNN queries and secure location perturbation. Thus, for this step, the secure protocol $\pi^{g|f}$ is due to the privacy guarantee of (ϵ, δ)-Geo-Indistinguishability (Geo-I) [12]. The security analysis for the other steps is similar to that for asymmetric radius-unknown queries.

A.3 Case Study on Semi-Honest Attack

Case Study. We have also conducted a case study to show how Hu-Fu defenses against the semi-honest adversaries when processing asymmetric federated range counting.

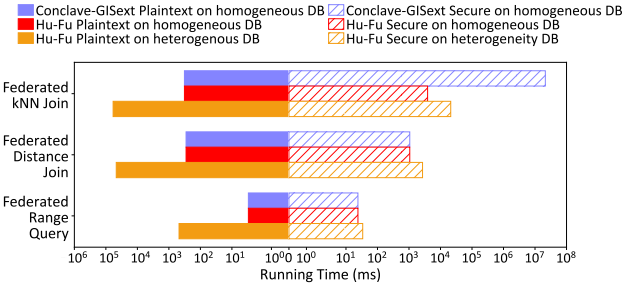


Fig. 18: Running time breakdown.

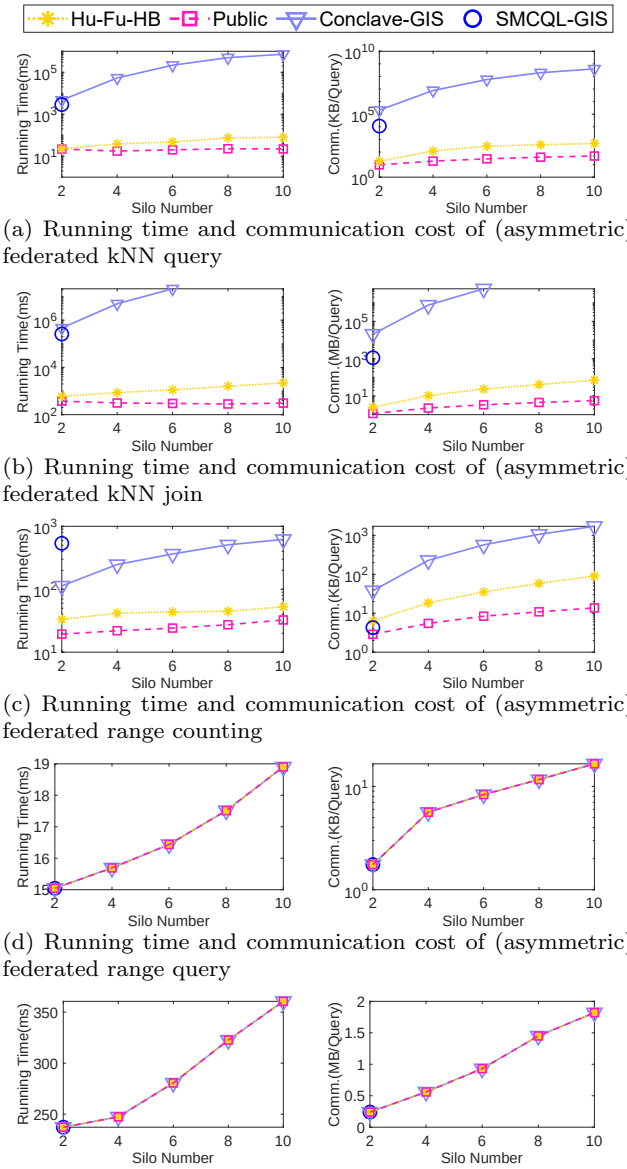
– **Experimental Setup.** The case study is based on the BJ dataset with 6 silos (*i.e.*, our default setting) and we vary the number m of colluded adversaries (*i.e.*, silos) from 1 to 3. Then, when processing asymmetric federated range counting, the attack will occur at the only secure operator (*i.e.*, secure summation [23]). To attack the secure summation, we use the method introduced in [23]. In general, a secure operator successfully defenses against the attack, if adversaries cannot get any feasible result after a long time (*e.g.*, two months in our experiment). Furthermore, the other experiment parameters are the same as those in Sec. 7.2.

– **Experimental Result.** As shown in Table 5, even when 50% of the silos collude, they cannot infer any feasible result even after two months' cracking. Note that it is commonly assumed in existing work that the portion of colluded adversaries is less than 50%. Based on the computational security [26, 40, 52], this result proves Hu-Fu is hard to attack.

B Additional Experiments on Heterogeneous Data Silos

Fig. 18 shows the results of asymmetric federated range query, distance join, and kNN join in the experiment of heterogeneous silos. We can first observe that our Hu-Fu can support all six spatial database systems, including PostGIS [9], MySQL [7], SpatiaLite [10], Simba [64], GeoMesa [31], and SpatialHadoop [21]. In addition, for all federated spatial queries in Fig. 18, we observe the running time of Hu-Fu becomes longer, compared with the results of homogeneous silos (*i.e.*, PostGIS in this test). This is because the time cost of Hu-Fu's plaintext primitives (*i.e.*, plaintext range query) increases due to the slowest spatial database system. We can also observe that the experimental patterns of federated distance join and federated kNN join are similar to those of federated range query and federated kNN query. This is because a federated distance/kNN join is decomposed into a series of federated range/kNN query. Besides, we also recommend that all silos can use more efficient spatial database systems to further speed up the federated spatial queries.

This observation also holds for processing symmetric federated spatial queries, since the query processing procedure also involves plaintext primitives such as the plaintext range query and kNN query.



(a) Running time and communication cost of (asymmetric) federated kNN query

(b) Running time and communication cost of (asymmetric) federated kNN join

(c) Running time and communication cost of (asymmetric) federated range counting

(d) Running time and communication cost of (asymmetric) federated range query

(e) Running time and communication cost of (asymmetric) federated distance join

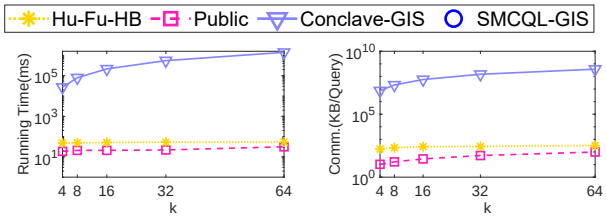
Fig. 19: Hu-Fu with an honest broker varying the silo number.

C Experiment on Hu-Fu with an Honest Broker

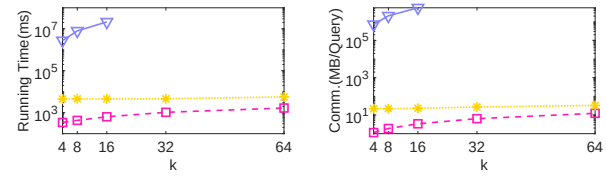
In the following, we conduct an experiment to demonstrate the performance of Hu-Fu with an honest broker (denoted by "Hu-Fu-HB").

Experimental Setup. This experiment has tested all asymmetric federated spatial queries on the real-world dataset BJ. We compare Hu-Fu-HB with SMCQL-GIS/Conclave-GIS in query processing efficiency measured in running time and communication overhead. Besides, the other experiment settings are the same as those in Sec. 7.2.

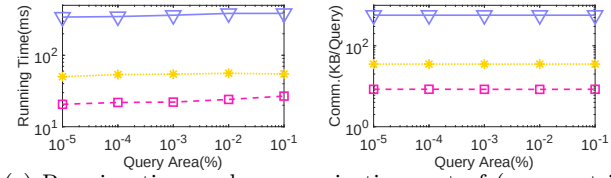
Experimental Result. Fig. 19 and Fig. 20 present the experimental results of Hu-Fu-HB and SMCQL-GIS/Conclave-



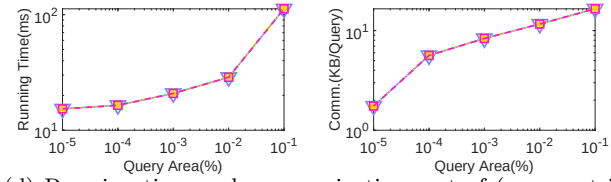
(a) Running time and communication cost of (asymmetric) federated kNN query



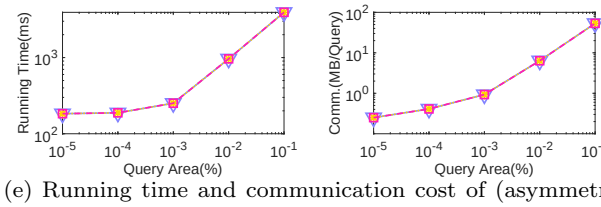
(b) Running time and communication cost of (asymmetric) federated kNN join



(c) Running time and communication cost of (asymmetric) federated range counting



(d) Running time and communication cost of (asymmetric) federated range query



(e) Running time and communication cost of (asymmetric) federated distance join

Fig. 20: Hu-Fu with an honest broker varying the query-specific parameter.

GIS in terms of running time and communication cost. First, We can observe that Hu-Fu-HB performs the same as SMCQL-GIS and Conclave-GIS on asymmetric federated range query and distance join. This is because all the compared algorithms (including ours) are simplified to Public due to the honest broker who collects the partial results (*i.e.*, sensitive data) of plaintext range query in each silo and is assumed to never reveal them to anyone else. Second, as for federated range counting, kNN query and kNN join, Hu-Fu-HB still outperforms SMCQL-GIS and Conclave-GIS with up to 4 orders of magnitudes faster in running time and 5 orders of magnitudes lower in communication cost. This is because our query writer is more effective to process queries on large-scale spatial data.

Table 6: Percentage of data of each country in *OSM country*.

Country	Percentage of data
China	21.7%
India	29.8%
Japan	39.4%
Malaysia	4.2%
North Korea	1.2%
South Korea	3.7%

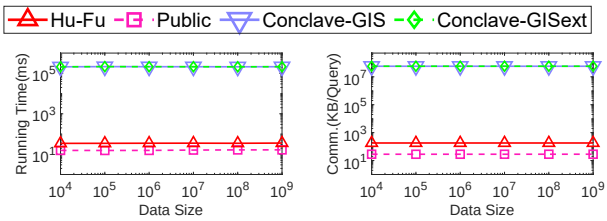
Summary. The experimental results demonstrate that compared to SMCQL-GIS and Conclave-GIS, Hu-Fu-HB exhibits similar efficiency in performing asymmetric federated range queries and distance joins. Additionally, Hu-Fu-HB achieves better efficiency in asymmetric federated range counting, kNN queries, and kNN joins. Although this experiment primarily focuses on asymmetric federated spatial queries, the observed patterns also apply to symmetric federated spatial queries. Specifically, the efficiency of symmetric federated range queries, distance joins, kNN queries, and kNN joins can be improved when an honest broker is present and a secure set union can be avoided. However, in contrast to these operations, symmetric federated range counting is not affected by the presence of an honest broker, as it does not involve a secure set union operator in its decomposition plan (as shown in Table 3).

D Experiment on Geographically Partitioned Dataset

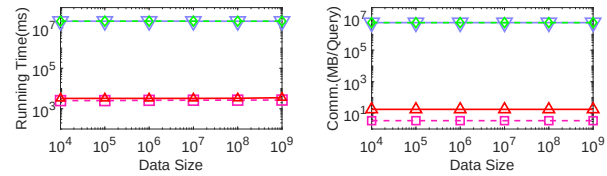
To explore the performance of Hu-Fu in the dataset where each silo corresponds to a single country, we have conducted an experiment on a new synthetic dataset (denoted by "*OSM country*").

Experimental Setup. The *OSM country* dataset consists of spatial points from six countries in OpenStreetMap [8]. These countries include China, India, Japan, Malaysia, North Korea and South Korea, and each of them corresponds to a single data silo. As shown in Table 6, the volume of data in each silo follows the native data proportion of the corresponding country in OpenStreetMap. For example, the numbers of data points in China, India and Japan are much larger than those of Malaysia, North Korea and South Korea. Here, we vary the total number of all data points from 10^4 to 10^9. The query workloads are mainly asymmetric federated spatial queries. Besides, the other experiment parameters are the same as those in Sec. 7.4.1.

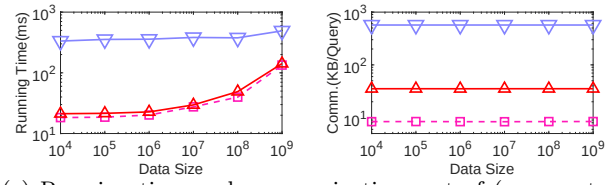
Experimental Result. The results of this experiment are shown in Fig. 21. First, we can observe that in the *OSM country* dataset, Hu-Fu also shows a significant improvement in both running time and communication cost over Conclave-GIS and Conclave-GISext on federated kNN query, kNN join and range counting. For example, Hu-Fu is at least 3 orders of magnitude faster than Conclave-GIS and Conclave-GISext in federated kNN query and kNN join, and costs 5 orders of magnitude lower network communication. Second, Hu-Fu achieves the same efficiency as Conclave-GISext on federated range query and distance join. Overall, the ranking of the compared solutions in terms of either running time or communication cost is consistent with the ranking in Sec. 7.4. Note that we exclude Conclave-GISext from the results of



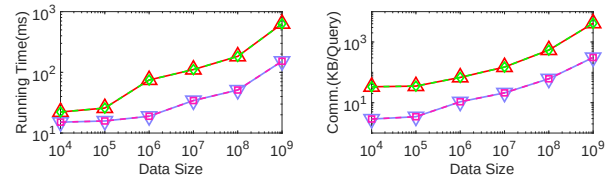
(a) Running time and communication cost of (asymmetric) federated kNN query



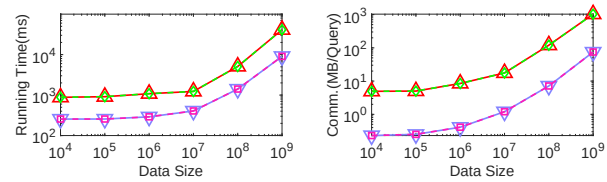
(b) Running time and communication cost of (asymmetric) federated kNN join



(c) Running time and communication cost of (asymmetric) federated range counting



(d) Running time and communication cost of (asymmetric) federated range query



(e) Running time and communication cost of (asymmetric) federated distance join

Fig. 21: Scalability test on synthetic dataset partitioned by countries.

federated range counting as shown in Fig. 21c, because this query does not need the secure set union to protect data ownership in this query.

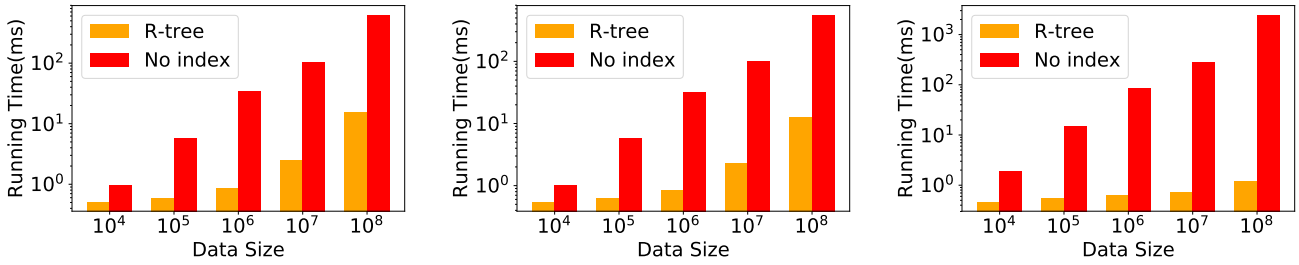
Summary. In this evaluation, Hu-Fu outperforms SMCQL-GISext and Conclave-GISext in terms of efficiency for asymmetric federated kNN queries, kNN join, and range counting. Meanwhile, Hu-Fu performs comparably to SMCQL-GISext and Conclave-GISext for asymmetric federated range queries and distance join. Thus, we can conclude that the experiment conducted on the *OSM country* dataset yields similar results to those observed in Sec. 7.4.1. This similarity may stem from the fact that none of the existing methods have leveraged the data distribution in their query processing strategies. Similarly, this new data partition is unlikely to alter the overall performance ranking of symmetric federated spatial queries.

E Experiment on the Improvement by the Index in Each Data Silo

To demonstrate that the federated spatial queries have already been accelerated by local indexes in Hu-Fu, we have conducted a new experiment on the efficiency improvement caused by these local indexes (*i.e.*, R-trees in our default setting).

Experimental Setup. In this experiment, we test the running time of the plaintext spatial queries in Hu-Fu (*i.e.*, plaintext range query, plaintext range counting and plaintext kNN query) on PostgreSQL with and without an R-tree. And the plaintext spatial queries are processed on the OSM dataset. Besides, we vary the data size from 10^4 to 10^8 and use the default setting of query area (0.001%) and k (16). The other experiment settings are the same as those in Sec. 7.1.

Experimental Result. As shown in Fig. 22, local indexes can improve the efficiency of plaintext spatial queries, especially when the data size is large. Specifically, the local index (R-tree) reduces the running time by up to 40×, 44× and 2042× when processing plaintext range query, range counting, and kNN query, respectively. Moreover, the plaintext spatial queries without local indexes cost even longer time than corresponding federated spatial queries in Hu-Fu. For instance, the running time of processing one federated kNN query by Hu-Fu takes 52 ms when the data size is 10^8, which is even faster than that of the plaintext kNN query without the local index (2465 ms). Thus, the experimental result proves that we have used local indexes in Hu-Fu to speed up the processing of the plaintext operators.



(a) Plaintext range query (b) Plaintext range counting (c) Plaintext kNN query

Fig. 22: Running time of plaintext spatial queries with/without R-tree.