Summary of Changes

Dear Editors and Reviewers,

We sincerely thank the Editors and the reviewers for their valuable and constructive comments and suggestions. The review comments are very helpful in further improving the quality of this work. Based on the suggestions, we have carefully revised the manuscript. Our detailed responses are as follows.

Yours sincerely,

Jie Zhu, Qi Li, Cong Wang, Xingliang Yuan, Qian Wang, and Kui Ren

Response to Associate Editor

Response: We thank the editor for paying great attention to this paper. Our response covers three parts. Each part answers in detail the questions or concerns raised by the individual reviewers. All sections and reference numbers below are based on the revised paper, and the major changes in the revised paper are colored in blue.

The major changes are in the following:

- We have revised the organization of this paper and moved the related work to **Section 2**.
- We have improved **Definition 3** to make it clearer and presented a high-level overview of our GSSE scheme in **Section 4.**
- We have updated our technical designs and analysis correspondingly with regarding to the revised yet polished **Definition 3.**
- We have significantly revised **Section 7** in terms of technical descriptions, and updated some experimental results according to our slightly polished design.
- We have carefully checked our writing, and tried our best to thoroughly improve the editorial quality of the paper, whenever possible.

Response to reviewer #1

Comment 1.1:

1) The authors use the term "the cloud" and "the clouds" without explaining what they mean exactly. I prefer the term "cloud service" instead of the term "cloud". "cloud can directly operating" could change to "cloud services can directly operate". In particular in the title, the authors should write "search in cloud services" instead of just "on cloud".

2) In Table 2, what do you mean by total number of the keywords set? Is it the size of the set?

3) In Section 3.2, the second parameter of the Prove operation is wrong (it contains a tau and a t symbol). Instead of < and > the authors should use \langle and \rangle.

4) Some grammar and spelling mistakes, such as "disclosure of privacy information -> disclosure of private information, temper -> tamper, exiting -> existing, the building proof index -> building the proof index, Comparision -> Comparison, Fig. 4 contains typos: exiting (twice) and bottem (twice)

Response 1.1: We appreciate the reviewer's kind suggestions on making our writing more precise, and we have made corrections and necessary changes accordingly.

1) In this revision, we have changed our title to "Enabling Generic, Verifiable, and Secure Data Search in Cloud Services", as suggested. We have also used the term "cloud service" instead of "cloud" throughout this paper.

2) For clarification, in **Table 2**, |W| means the total number of keywords in the keyword set, i.e., the size of the keyword set. We have made it clearer in this revision.

3) In **Section 4.2** (i.e., **Section 3.2** in our previous submitted version), we have fixed the mistakes in the *Prove*() algorithm, according to the comments by the reviewer.

4) We have corrected the grammar and spelling mistakes. We have also gone through the writing of this revision carefully so as to avoid such similar typos or grammar/spelling issues. Thanks again.

Comment 1.2: *What do you mean by always in the sentence "in particular when data are always updated by data owners"?*

Response 1.2: We thank the reviewer for raising this question. By "always", we mean the scenarios where data updates happen frequently. We have revised that sentence in our Introduction, i.e., **Section 1**, to make the description more precise. Kindly note that our GSSE system not only supports result verification but also can prevent data freshness attacks under data updates by leveraging a timestamp-chain mechanism. Therefore, compared to existing verifiable SSE schemes, such as those in [13] [14] [17] [18], which only verify search results over static data, our design has a great advantage of supporting data updates. This is especially desirable by many modern cloud storage applications when data update happens frequently.

Comment 1.3: *Still I think the benchmarks are not convincing because they are not executed in a cloud environment.*

Response 1.3: We thank the reviewer for raising this comment, and we would like to further elaborate about our evaluations below.

Our reported experimental results, shown in Fig. 5, 6, 7, 8, 9, 10, 11, 12, 13, and table 3, mainly focused on reporting the computation (operation execution latency) and communication overhead (bandwidth cost) of various operations in our GSSE design, and the extra storage cost required. To better illustrate our performance cost, we have reported the measurement with comparison to some state-of-the-art SSE designs in [11]. Some updated experiments due to a slightly polished design, as suggested by Reviewer #3, have also been added in this revision.

Note that all these measurements do not take into account the network transmission and propagation delay, which varies in different specific network contexts and is an orthogonal issue. We intentionally exclude the network delays in our performance evaluation because such delays have nothing to do with the essential extra cost directly introduced by our verification design. We do, however, have shown the communication overhead in terms of the message size. From these perspective, we believe our reported measurements have faithfully demonstrated the efficiency of our generic design, even if they are not measured based on a remote machine in cloud.

From above, we can see that our reported overhead measurements are only affected by the computing hardware and the chosen dataset. If we move our local testbed to some virtual machine in cloud, with the same hardware configuration, the reported experimental results shall remain almost similar without noticeable difference, if not exactly the same. But considering the cloud services usually have much better computing resources, we expect that the overall reported performance results (in particular computation overhead) should be much improved in a cloud environment. The communication overhead (extra bandwidth cost) in the cloud settings and the extra storage cost should be the same to that in our current settings.

Finally, we want to emphasize that even in our current experiment results, the additional cost incurred by our GSSE scheme is already very small, if not negligible. The overhead of the *Init*() algorithm in our scheme only takes 1.9% that of the entire initialization phase, the *Prove*() algorithm only takes 2% that of the entire search phase and the *Add* and *Delete* operations of the *Update*() algorithm only take 17% that of the entire update phase. Moreover, as shown in **Table 3**, the communication overhead introduced by GSSE is also below 10%.

Many of the above view points and our considerable amount of updated evaluation results are in our revised **Section 7.** Please refer to the newly updated **Section 7** for more details. Hopefully, it helps make our experiment results more convincing.

Response to reviewer #2

Comment 2.1: *In my view, the related work section should be presented in the first part of the paper, not at the end.*

Response 2.1: Thanks for the suggestion. In the revised paper, we have taken the suggestion and adjusted the organization accordingly, by moving the related work section right after the introduction section, i.e., **Section 2**.

Comment 2.2: *Experimental comparison with related system appears to limited as some extent since only one figure is presented to compare GSSE with Dynamic SSE. I suggest to extend comparison by providing additional performance figures with related systems.*

Response 2.2: We thank the reviewer for pointing out this issue. In addition to comparing the performance of our scheme with SSE scheme [11] in **Fig.13**, we have also compared the communication overhead, as in our **Table 3**. The average size of the search results of the SSE scheme [11] is about 53KB, while the proof size of our scheme is lower than 3KB. This means that the overhead introduced by GSSE is less than 6%. Moreover, the size of tokens used in the SSE scheme [11] is 390 Bytes, and the size of the search tokens in our scheme is only 32 Bytes. Thus, the additional overhead is less than 9%. All the measurements are the average values with 50000 runs. The execution delays of various operations shown in **Fig.13** further demonstrate that our scheme is feasible in practice. Besides, in this revision, we have presented a slightly polished design as suggested by Reviewer #3. Accordingly, we have also updated a number of experimental results, as reflected in the **updated Fig. 5, 11, 12**, so as to more accurately reflect the performance of our polished design. Please see our **response 3.4,** the updated **Section 4.2**, **Section 5.3**, and **Section 7** for further details.

Kindly note that our proposed GSSE aims to be generic and support all existing dynamic SSE schemes. Therefore, given any fixed dataset, the absolute value of the extra overhead by result verification in GSSE, such as the additional computation, storage, and communication cost, should remain the same, regardless of the underlying SSE schemes we choose to compare against. Our reported experimental results for the given dataset, shown in Fig. 5, 6, 7, 8, 9, 10, 11, 12, are just one concrete example to demonstrate that the exact overhead values are indeed not large. The concrete comparison of GSSE against the state-of-the-art dynamic SSE in [11], shown in **Fig. 13** and **Table 3,** is just another instantiation to show that the extra overhead in terms of percentage introduced on top of existing SSE schemes is still small. We hope our improved experimental results in this revision have addressed the comments satisfactorily.

Response to reviewer #3

Comment 3.1: *Editorial quality: Even after going a round of reviews, the paper still has a lot of grammatical mistakes and typos. I mean I understand grammatical mistakes can happen in few places but the number of problems is huge. It actually distracts the readers from following the discussion.*

Response 3.1: Many thanks for the kind comments. In the revised paper, we have carefully checked our writing, and tried our best to thoroughly improve the editorial quality of the paper, whenever possible. We have made sure that high readability is one of our top priorities in this revision.

Comment 3.2: *The way the authors have written the paper leaves the readers with questions which are not important. For instance, the authors write, "It can be applied to various SSE schemes [10], [11], [16] to provide search results verifiability for data users, ..." The obvious question the reader would face after reading this sentence is that are these the only SSE schemes that are applicable, or they can support any well-formed (for some notion of well-formedness) SSE schemes. Such sentences abound in the paper.*

Response 3.2: We thank the reviewer for pointing out this issue. We have thoroughly improved/polished our writing in this revision so as to avoid any confusion or misleading. Kindly note that our GSSE design provides generic verification for any SSE schemes, because fundamentally the verification index in our scheme is independent from the SSE construction and does not rely on any specific SSE instantiations. Therefore, our scheme can be applied to any searchable encryption schemes, including but not limited to those in [10][11][16]. As acknowledged in **Table 1,** we are not the first design that supports the generality of the verification in searchable encryption. Quite a few prior designs also have the generality feature, though they somewhat fall short in other selected comparison aspects. In our paper, we only refer to these three exemplary papers [10][11][16] because they are relatively notable and well-known SSE constructions. In the revised manuscript, we have clarified this issue in **Section 1**.

Comment 3.3: *Structure of the paper: The structure of the paper can be much more improved than its current situation. The main ideas are generating proof indices (considering updates), and* *freshness maintenance through timestamp chains. For explaining these two notions one actually does not need any concrete cryptographic schemes. The authors should be providing insights as to how it works, using possibly an abstract example. The authors explain everything with different cryptographic keys and constructs which distract the readers from getting the main insight. Explain the main insight first. It is not necessary to give details right away. Once you can explain this in the abstract format, then the authors can say, "recall we need a construct with these guarantees, one can implement such a construct with incremental hashing in the following fashion."*

Response 3.3: We thank the reviewer for the concrete suggestions. In order to better elaborate our algorithms, we have added a high-level overview in **Section 4** to clearly explain the structures of our solution in the revised manuscript. We have also presented an updated/polished version of the collection of GSSE algorithms in **Definition 3**, followed by our detailed algorithms in **Section 5** and abstract examples (**Fig. 3** and **Fig. 4**) to explain these algorithms. Since our GSSE approach consists of building the proof index and the timestamp-chain for result verification, we have put these two structures together in **Definition 3**. For clear explanation, we have carefully described them respectively in **Section 5**.

Comment 3.4: *Assumptions: This brings me to my final concern. The paper makes the following assumptions: (1) The three systems should be loosely time-synchronized; (2) The data users and cloud service provider cannot collude; (3) The data owner shares all three keys K1, K2, and K3 with the data users, i.e., data users are trusted. Although (1) and (2) are bad enough, I can let it go. The third assumption, however, is very restrictive. If the data owners and users share the same three symmetric keys, what is stopping the users to add new stuff to the cloud storage? The data users have all the three keys K1, K2, and K3 to call PreUpdate after which they can call update; completing the update. One can say the user does not need access to K1, K2, and K3. Without accessing K1, K2, and K3, however, the users cannot verify the search result. I have tried to follow the scheme closely and I realized that this is straightforward cryptographic application due to the assumption. I would suggest the authors investigate a case where the users do not have the capability to change the cloud storage without owner authorization.*

Response 3.4: We thank the reviewer for raising these comments. We have made necessary changes in our updated / polished **Definition 3** on GSSE to make our technical description more precise and rigorous and also to address the raised concerns. Among others, our change includes making explicit assumption about the authenticity of the output generated by the *Init*() and *PreUpdate*(), the two algorithms that can only be called by data owner to build and update the index.

Specifically, we have added explicit digital signing key pair (ssk, spk) to the output of *KGen*(), the probabilistic key generation algorithm run by the owner. The owner will be using the secret key ssk to sign the authenticator $\pai$, whenever he calls the *Init*() and *PreUpdate*() to build or update the index. Correspondingly, users will need to explicitly use spk to verify the owner's digital signature before conducting any subsequent result verification operations, as shown in our updated *Verify*() algorithm. Due to the slight polished **Definition 3** on GSSE, we have also made changes accordingly in the algorithm descriptions and technical descriptions in **Section 4** and **5**. A considerable amount of experimental results in **Section 7, Fig. 5, Fig. 11, and Fig. 12,** are also updated accordingly, together with the technical descriptions. As a result, users still need access to K1, K2, and K3 to conduct verification. But the authenticity checking of the $\pai$ value using spk is a pre-condition. Since only the data owner has the signing key ssk to sign the authenticator $\pai$, we have made it more clear that users will not be able to call *Init*() or *PreUpdate*().

As an additional remark, in our paper, we assume that the data owner is trusted and the data owner will assign three keys K1, K2, and K3 to the data users he/she trusts. Authorized users can surely use these keys for search and verification. This scenario is quite common in the literature of searchable encryption. In fact, there are four types of architectures in Searchable Encryption: single writer/single reader (S/S), single writer/multi-reader (S/M), multi-writer/single reader (M/S) and multi-writer/multi-reader (M/M). What we have described in this paper is the S/M architecture, since the access privilege of users can be well controlled by fine-grained access control mechanisms, such as role-based access control policies. The data owner can assign different roles for his/her users based on their responsibilities. Each role corresponds to an access privilege, such as read-only. There are many existing literatures that studied how to enforce the user authorization via one-to-many encryption schemes, like broadcast encryption (BE) or attribute-based encryption (ABE) to control the sharing of these secret keys. In our **Section 8**, we have presented one well-known instantiation on using BE to control the key sharing by data owner towards multiple users. We thank the reviewer again for giving us the opportunity to polish our paper. We hope our revision together with the above explanation will help address the raised concerns.

Comment 3.5: *Some of the algorithm listings are not very useful, e.g., Verify. The authors can state that one calls the check function to verify the authenticator and then only calls the generate function. Right now it is cryptically written inside the text in Section 3.2. Algorithm listing 1 does not take K3 as an argument whereas it is mentioned in the signature in Section 3.2 Few of the function signatures in Section 3.2 are inconsistent.*

Response 3.5: Thanks for the comments. By following the suggestion, we have simplified our description in **Definition 3**. *Check*() and *Generate*() algorithms are necessary parts of the verification process, i.e., the *Verify*() algorithm. Without them, the *Verify*() algorithm will have to be incomplete. Therefore, we put them together in **Definition 3**. Moreover, since GSSE achieves the goal of verifiable SSE by combining the proof index and the timestamp-chain, **Definition 3** also puts these two structures together. In **Section 5**, in order to explain our GSSE solution in detail, we have described these two structures and the corresponding algorithms respectively. We have modified the *Init*() algorithm and made it consist with **Definition 3** in **Section 4.2**. Since the *Init*() algorithm described in **Section 5.1** only emphasized on how to build the proof index, we have delayed the detailed description of generating the authenticator $\pai$ to **Section 5.3**. We have clarified all these issues in **Section 5**, in addition to the newly added explicit assumption of the digital signature signing keys we have just explained in our **Response 3.4**.

References

[10] S. Kamara, C. Papamanthou, and T. Roeder, “Dynamic searchable symmetric encryption,” in Proc. of CCS, 2012, pp. 965–976.

[11] D. Cash, J. Jaeger, S. Jarecki, C. S. Jutla, H. Krawczyk, M.-C. Rosu, and M. Steiner, “Dynamic searchable encryption in very-large databases: Data structures and implementation,” Proc. of NDSS, 2014.

[13] K. Kurosawa and Y. Ohtaki, “Uc-secure searchable symmetric encryption,” in Proc. of International Conference on Financial Cryptography and Data Security (FC), 2012.

[14] Q. Chai and G. Gong, “Verifiable symmetric searchable encryption for semi-honest-but-curious cloud servers,” in Proc. of International Conference on Communications (ICC), 2012.

[16] E. Stefanov, C. Papamanthou, and E. Shi, “Practical dynamic searchable encryption with small leakage,” in Proc. of NDSS, 2014.

[17] R. Cheng, J. Yan, C. Guan, F. Zhang, and K. Ren, “Verifiable searchable symmetric encryption from indistinguishability obfuscation,” in Proc. of AsiaCCS, 2015.

[18] W. Ogata and K. Kurosawa, “Efficient no-dictionary verifiable sse,” IACR Cryptology ePrint Archive, vol. 2016, p. 981, 2016.