IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 24, 2021, accepted June 17, 2021, date of publication July 2, 2021, date of current version July 13, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3094188

Improving Privacy in Data Service Composition

GYAN PRAKASH TIWARY[1], ELENI STROULIA[2], (Member, IEEE), AND ABHISHEK SRIVASTAVA[1]

[1]Discipline of Computer Science and Engineering, Indian Institute of Technology Indore, Simrol, Madhya Pradesh 453552, India
[2]Faculty of Science-Computing Science, University of Alberta, Edmonton, AB T6G2R3, Canada

Corresponding author: Abhishek Srivastava (asrivastava@iiti.ac.in)

This work was supported in part by the Ministry of Electronics and Information Technology (MeiTY), Government of India, and in part by the Science and Engineering Research Board (SERB), Government of India.

ABSTRACT Today, the de-facto standard mechanism for data service providers to share their services is through web-service interfaces; clients invoke the service through request messages and receive the data as payloads in the corresponding response messages. Typically, clients need information beyond what any single provider offers; in such cases, multiple data services must be composed to provide a complete solution meeting the client needs. The term ''data-service composition'' refers to a unified interface that delivers to the client the data it needs in response to a single request message, as if it were available withing a single source. Data-service composition is useful and convenient for the requesting client, but raises privacy concerns since a participating data-service provider potentially can infer information about the data held by other providers. In this paper, we propose a data-service composition method that relies on a mediator for the communication between any two service providers, ensures that the mediator is strictly following the data-composition plan, and maintains privacy between the mediator and service providers. The data service provider first authenticates that the input data is coming from the correct source as per the composition plan, and this is done whilst ensuring complete privacy between the mediator and other service providers. Similarly, data service providers also authenticate the destination of their output data. The approach is validated and its performance evaluated using a real world online retail dataset.

INDEX TERMS Web services, privacy-preserving service composition, privacy-preserving web data integration, privacy in data service composition.

I. INTRODUCTION

Data of various types is available in a variety of domains, including medical databases, retail databases, demographic databases, and others. Clients can access and use this data, based on their specific application requirements, through web-service interfaces provisioned by the data owners. Typically, complex applications require information beyond what any single provider may own and offer; in such cases, multiple data services must be composed to provide a complete solution meeting the client needs. The term ''data-service composition'' refers to a unified interface that invokes multiple data providers and synthesizes their data to deliver to the client all the data it needs in response to a single request message, as if it were available withing a single source.

Srivastava *et al.* [18] formally define a ''Service Composition Plan'' as: ''an arrangement of the web services in the query into a directed acyclic graph (DAG) $\mathcal{H}$ with parallel dispatch of data denoted by multiple outgoing edges from a single web service, and rejoining of data denoted by multiple incoming edges into a web service''. Each query has its own service composition plan. A directed edge e_{ij} from a service provider *DSi* to *DSj* establishes *DSi* as the parent of *DSj* and implies that, while the plan is executed in response to a query, *DSi* precedes *DSj* and that *DSj* consumes the data produced by *DSi*.

An example service composition plan is diagrammatically depicted as a directed acyclic graph $\mathcal{H}$ in in Figure 1. In Figure 1, there are four data service providers (*DS1*, *DS2*, *DS3* and *DS4*) that provide details of a retail store. If a client requests information on ''how many lunchboxes were purchased by people belonging to France on December 3, 2012'', the execution of this query requires the involvement of all the four data service providers. The service composition plan for the query is shown in the Figure 1. *DS1* is invoked first with a ''Lunch Box'' description to retrieve

The associate editor coordinating the review of this manuscript and approving it for publication was Emanuele Bellini.

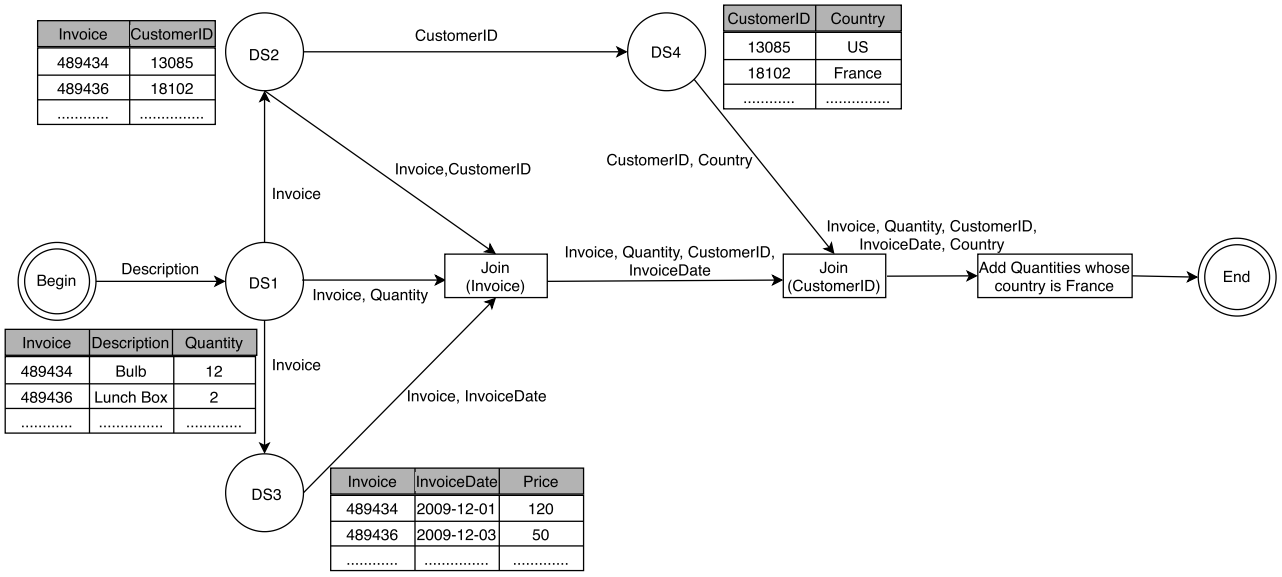

FIGURE 1. Service composition plan (data flow between two services always happens through mediator) .

the ''Invoice'' and ''Quantity'' of the purchases. Next, *DS2* and *DS3* are invoked with ''Invoice'' as the input to obtain ''CustomerID'' and ''InvoiceDate'' for the invoices corresponding to ''Lunch Box''. The outputs from *DS1*, *DS2* and *DS3* are joined together to produce a table *T1(''Invoice'', ''Quantity'', ''CustomerID'', ''InvoiceDate'')*. *DS4* is then invoked with ''CustomerID'' as the input and results in *T2(''Custo-merID'', ''Country'')*. T1 and T2 are joined to get *T3(''Invoice'', ''Quantity'', ''CustomerID'', ''InvoiceDate'', ''Country'')*. Finally, the sum of the ''Quantity'' value in table T3 that corresponds to ''Country'' as France gives the final result to be returned to the invoking client.

The service-oriented architecture style dictates that providers, involved in a composition for the purpose of fulfilling a client's needs, should not be aware of each other. Furthermore, data service compositions impose additional privacy-related constraints that further enforce this requirement [3]. More specifically, a service provider may encrypt the data it contributes to the service composition, in order to prevent other participants to gain any information. Alternatively, a service provider may allow the sharing of some attribute values, except in the case of specific queries. Furthermore, a service provider may disallow the sharing of some attribute values with some specific service providers participating in the service composition plan. To support these requirements, it is essential that the participating service providers have a right to know the service composition plan, which should be shared before the execution of the plan.

Privacy is an important concern in data service composition: a service provider *DSi*, participating in a service composition plan, should be able to choose whether to hide one or more of its attributes from all, or some of, the other service providers in the composition plan. An attribute that a data service provider does not want to share is called a ''privacy critical attribute''.

Privacy is a major challenge to the smooth execution of a service composition plan. For example, in Figure 1, if ''Invoice'' is a privacy critical attribute of *DS1*, then this implies that *DS1* does not want to share its values with anyone else. However, according to the service composition plan, *DS1* needs to connect with service *DS2* and share the relevant invoice number (489436 in this example) in order to get the corresponding ''customer ID'' related to the lunchbox purchased with the above invoice. This sharing would be a direct infringement of the privacy of *DS1*, which raises the question: how can this service composition plan be executed?

An approach to address the issue of privacy in data service composition is through the use of privacy models such as K-Anonymity [4], and/or its variants such as L-Diversity [5] and T-Closeness [6]. K-Anonymity is explained as follows: in a service composition plan where *DSi* is the parent of *DSj*, a query is executed on *DSj* using as input information produced aa output of *DSi*. In such a situation, to preserve the privacy of *DSi*, the K-Anonymity approach prescribes that instead of querying for just the one attribute value of *DSj* corresponding to the output attribute value of *DSi*, *K* attribute values of *DSj* are queried. These *K* attributes include the value corresponding to the output of *DSi* and *K-1* other attributes. The parent (i.e., *DSi*) determines the value of K.

K-Anonymity can be explained using the example shown in Figure 1. *DS1 is queried for the ''Invoice'' of a lunchbox (invoice number 489436). Subsequently, database DS2 is queried for the ''CustomerID'' of the same ''Invoice'' (i.e., 489436). Conforming to the K-Anonymity approach, a query on the data service provider DS2 is not just executed for invoice number 489436 which is of interest here, but for K different values where K>1. For example in Figure 1, when K is 3, queries on DS2 are executed for* 3 *different values of the attribute ''Invoice'' (say 489435, 489436 and 489437), among which the actual invoice of interest,489436,*

is included. All the three values of "invoice" are such that they are present in the DS2 database. In such a scenario, therefore, DS2 knows that a certain value belongs to DS1 only with a probability of 1/K (in this example 1/3 as K is 3). DS1, that needs to protect the privacy of its "Invoice" data, determines the value of K. K-Anonymity is also referred to as K-Protection or the Value Generalization Method. Several research methods have been developed relying on K-Anonymity for privacy preservation and are discussed in greater detail in Section II.

Executing a data service composition involves several tasks such as, selection of service providers, development of the service composition plan, invocation of queries on the various data service providers, storing the intermediate query results, performing join operations on data received by different providers, and maintaining the privacy of service providers through value generalization. Responsible for the above tasks is a special service, called the mediator. Mediators can be trusted or untrusted. If a mediator is trusted, it can be used as a warehouse to place the intermediate results of the query in plain form. If a mediator is untrusted the providers' data is encrypted before it is stored in the warehouse. If the mediator is untrusted, there is little or no expectation that it should contribute towards privacy preservation. Most research today on mediators is towards improving the working of untrusted mediators.

Managing encrypted intermediate data is a major challenge with untrusted mediators because the mediator needs to handle encrypted intermediate data in queries or join operations without seeing the content. To overcome this, Barhamgi *et al.* [13] developed OPES [14], an encryption method for numerical data in a manner that preserves the order of the encrypted values. This means that the comparison of two encrypted values will produce the same result as the comparison of the corresponding decrypted values. Barhamgi *et al.* [13] use OPES for query execution, join operations, and deciding on the value of K for K-Anonymity. A major limitation with OPES is that it can only be used to encrypt numerical data. Using OPES, therefore, a service provider is able to secure only its numerical privacy-critical attributes. In this paper, we propose to overcome this limitation by making use of hashing. Each parent-child pair in a service composition plan shares a secret string unknown to the mediator. This secret string is appended to each value of the privacy critical attributes and its hash (e.g., SHA-256) is calculated (Section III). As the secret string is the same for a parent-child pair, the hash of a certain value will be the same for both service providers. In this way, the hash values of non-numerical data can also be compared lexicographically by the mediator without seeing their actual values.

Another major limitation of existing data service composition approaches is the assumption that the mediator is adhering to the service composition plan. Service providers do not, therefore, authenticate each other. They do utilise value generalizations (mostly using K-Anonymity) to protect data from each other by encrypting and sharing data with the mediator using OPES [14], but do not check whether the mediator is adhering to the shared composition plan. It is therefore possible that a mediator can share a false service composition plan and introduce a fake service provider in the composition. If a service composition, for example, consists of four services *DS1*, *DS2*, *DS3*, and *DS4*, and $DS1 \rightarrow DS2 \rightarrow DS3 \rightarrow DS4$ is the actual service composition plan. An unethical mediator can invoke another service provider *DS5* into the composition about which the remaining four service providers are unaware. Alternatively, an unethical mediator can also surreptitiously change the shared service composition plan (e.g. $DS1 \rightarrow DS4 \rightarrow DS2 \rightarrow DS3$) without the knowledge of *DS1*.

In this paper, we describe a method for supporting data-service composition through an untrusted mediator, as follows.

1) During data service composition, the data of one service provider needs to be protected from other service providers. The mediator facilitates the maintenance of privacy between service providers in addition to taking on various responsibilities such as service selection, creation of the service composition plan, querying, joining, and communication between service providers. The mediator is untrusted and is not allowed to see any data; therefore, all data exchanges are encrypted.
2) Only numeric data can be secured using the current standard of OPES [14]. We propose concatenation of a secret string with data followed by hashing. Therefore, privacy critical attributes in our case do not necessarily need to be numeric. The proposed approach enables the mediator to perform operations such as join, query execution, or value generalization using K-Anonymity without seeing the actual data values.
3) It is the right of every service provider to review and agree on the service composition plan before participating in a composition. The service providers should be able to validate that they are participating only in the composition plan to which they have agreed. The service providers should be able to validate that their input data is originating from their parent service provider and that their output data is sent to their children service providers, based on the agreed upon composition plan.

The remainder of this paper is structured as follows. Section II describes existing research endeavours in data service composition. Section III provides details of the approach for effective privacy preservation in data service composition proposed in this paper. Section IV explains the approach adopted for evaluating the efficacy of the proposed technique. Finally, Section V concludes the paper.

II. RELATED RESEARCH

Data integration systems have always been a major topic of research [2] and maintaining the privacy of the data sources involved has always been a challenging task in this area [7]. As discussed in Section I, mediator-based solutions are

typical for data service composition, assigning the overhead activities related to the composition to the mediator.

In Yau and Yin [8], the mediator acts as a data repository. The data useful for the query is collected by the mediator from different data service providers in hashed format. The collection of these data in hashed format prevents data from being reused in another query. However, this technique does not prevent information leaks between different data service providers.

Tbahriti *et al.* [3] have coined terminologies like ''Privacy Policy'' and ''Privacy Requirement''. A data provider's privacy policy dictates what operations the data service provider can perform on its own data; the privacy requirement of a data service provider guides how the data provider wishes others to use its data. Through their privacy requirements, data service providers exercise their the right to conceal their data (i.e., output). Two data service providers may collaborate with each other in a composition only when the privacy requirement of the source data service provider is compatible with the privacy policy of the destination data service provider. A formal privacy model is defined to allow a service to define a set of privacy policies and requirements. A mediator establishes the compatibility between the privacy requirement of the source and the privacy policy of the destination services, or else the service composition is impossible. To that end, the mediator negotiates between source and destination service providers to ease their privacy policy and privacy requirement. The techniques proposed by [8] and [3] rely on centralized entity that can be trusted by all the data service providers in the data integration plan.

Fung *et al.* [9] and Mohammed *et al.* [10] create an atomized view of data using K-Anonymity. However, in their approach the service provider and the service consumer can know the data they have in common with each other, which is not desirable. There are some other methods (i.e., [11] and [12]) using K-Anonymity, but they all lack somewhere in efficiency and cost effectiveness.

In the work of Barhamgi *et al.* [13], the mediator is not considered trustworthy in the technique proposed by [13]. Data service providers encrypt their data before sharing it with the mediator and the mediator performs join on the encrypted data. Their Order Preserving Encryption Scheme (i.e., OPES) algorithm [14] encrypts numeric data values such that they can be compared without decrypting them. In this method, a K-protection mechanism prevents data leakage between two service providers, which is an improved versions of k-Anonymity and Private Information Retrieval (PIR) [15]. PIR provides a primitive for accessing outsourced data from a server by a client while preventing a server to learn about client's access pattern. An important shortcoming of this method is that only numerical values are encrypted here. Nothing has been said explicitly about what to do if the identifier attributes are not numerical.

In none of the above-mentioned related research do providers authenticate the origins of their data inputs, against the composition plan, which implies that an unethical mediator may not follow the shared service composition plan. The mediator can introduce a foreign data service provider in the service composition plan without consent of other service providers in service composition plan. Addressing this fundamental shortcoming of the state of the art is a key objective of our work.

III. PROPOSED METHOD

In this paper, we use a mediator-based approach for data service composition. We consider the mediator to be untrustworthy. Intermediate data is shared with the mediator in a secure manner such that the mediator can orchestrate the data composition plan, perform join operations on the collected data without accessing the data. Our approach is similar to that of Barhamgi *et al.* [13] but our method considers privacy-critical attributes of any type, not only numerical. Similar to [13], our methods uses K-Anonymity, more specifically the *"Dataset-based identifier generalization"* method discussed in [13], for value generalization to preserve the privacy of data service providers during data service composition. In addition, our approach also ensures authentication of the intermediate input and output data during data service composition. The service providers participating in the composition plan authenticate their respective parent(s) (the preceding data service in the composition plan) and child(ren) (the succeeding data service in the composition plan). The proposed approach also ensures that the mediator strictly adheres to the shared service composition plan and does not introduce illegitimate, external service providers.

In spite of being 'untrustworthy', it is the mediator's responsibility to orchestrate the data service composition. A client application queries the mediator and the mediator returns privacy sanitized results, without gaining access to intermediate privacy critical data. Algorithm 1 describes the mediator's role in detail in data service composition; and Algorithm 2 describes the part that the service providers play. The detailed steps followed by the mediator during the Service Composition are described in detail in the following subsections.

A. MEDIATOR GENERATES AND SHARES SERVICE COMPOSITION PLAN

It is the responsibility of the mediator to prepare a service-composition plan to execute the client query (Steps 3 and 6 of Algorithm 1, Steps 1 and 2 of Algorithm 2). The mediator maintains a mapping table to keep a record of services along their properties such ''service name'', ''input'', ''output'', and the like (Step 1 of Algorithm 1). The mediator next selects a set of services from the mapping table to execute the requested query. The selection of services is based on their properties such as input, output, and content. The input query is broken into several sub-queries and the sub-queries are assigned to different service providers. This process is called query rewriting (Step 12 of Algorithm 1). In this paper, our focus is only on improving privacy in data service composition and we do not dwell upon service

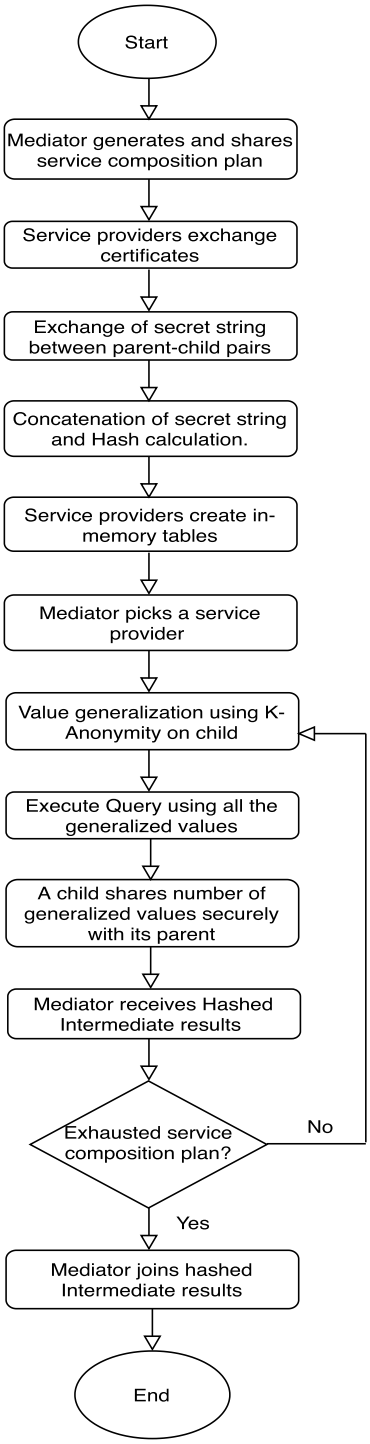

FIGURE 2. Steps in privacy preserving data service composition.

selection and query rewriting. We adopt these from [16], [18], and [17].

Subsequent to service selection and query rewriting, the mediator decides the order of execution of sub-queries on the selected services. This ordering forms a directed acyclic graph of data service providers called the Service Composition Plan (Figure 1). An edge between two nodes *DSi* and *DSj* (nodes represent the service providers) in the Service Composition Plan indicates that the output of *DSi* is an input to *DSj*. For example, *In Figure 1 the input to DS2 and DS3 is the output of DS1*. We adopt the composition plan generation technique from Srivastava *et al.* [18]. The mediator shares the Service Composition Plan with the selected data service providers (Step 6 of Algorithm 1).

B. SERVICE PROVIDERS EXCHANGE CERTIFICATES

Service providers in a service composition plan exchange their respective authentication certificates with each other through the mediator (Step 7 of Algorithm 1, Step 2 of Algorithm 2). The mediator requests the X.509 certificates of all service providers in the composition plan [19] which is one of the possible certification mechanisms to authenticate the service providers in the composition plan. The mediator shares the certificate of each service provider with all the other service providers i.e. the certificate of each service provider *DSi* is shared with service providers *{DS1, DS2, ……., DSn}-{DSi}*.

C. EXCHANGE OF SECRET STRINGS BETWEEN PARENT-CHILD PAIRS

All the parent-child pairs in the service composition plan share a common secret string between them (Steps 8, 9, and 10 of Algorithm 1, Steps 4, 5, 6, and 12 of Algorithm 2). The mediator facilitates the exchange of the secret strings between them, without it seeing the secret string. To create the secret string each parent service provider chooses a random secret string and encrypts it using the public key of its respective child(ren). This encryption is followed by another encryption with its own private key (i.e., $PRI_{parent}(PUB_{child}(SecretString))$). This doubly encrypted secret key is then passed on to the mediator that distributes it to the parent's child(ren). If a service provider happens to be playing the role of both a parent and child in the composition plan, it is associated with two secret strings: one that it receives from its parent and the other that it shares with its child(ren).

D. CONCATENATION OF SECRET STRING AND HASH CALCULATION

All service providers append the secret strings of their respective parents to the entries of the privacy critical attributes (Step 13 of Algorithm 2). Next, an SHA-256 hashing of each value of the privacy critical attribute appended with the secret string is created. The same hash string is created for values that are present with the both the parent and child of the parent-child pair. The mediator does not know the attribute values nor the concatenations of the values and the secret string, but it is able to compare or execute join operations on the hashed values. Every new data service composition has a new secret string between a parent-child pair.

E. SERVICE PROVIDERS CREATE IN-MEMORY TABLES

Each service provider maintains a table for its privacy-critical attributes (Steps 14 and 15 of Algorithm 2) and a second table

comprising the ''*hashed-value*'' and the actual '*value*'' of the privacy critical attributes. The *hashed-value*, as mentioned earlier, is created by concatenating the secret string from the parent with the value of the attribute and following it with applying SHA-256 to it. This additional table for hash values ensures that the original table is untouched. For example, *in Figure 1 DS2 maintains an in-memory table comprising T(Invoice-Hash, Invoice)*. The table remains sorted (lexicographical sorting) with respect to the hashed-value (i.e., ''InvoiceHash'' in the case of DS1). When the mediator needs to execute a query for a value, it executes the same with the help of the hashed values (this is because the mediator only has hashed value). The service provider is able to know the actual value of the attribute with the help of tables in the memory and executes the query. There are several advantages of using an in-memory tables. First, the original table need not be sorted. Second, a service provider can maintain different in-memory tables for different composition plans. Different in-memory tables have hash values created, each with a different secret string. Third, the in-memory table protects our method from the privacy breech mentioned in detail in Section III-G.

The use of the secret string is explained through the following example. In Figure 1, *DS1* encrypts a secret string, $SecretString_{DS1-DS2-DS3}$, with the public keys of *DS2* and *DS3* and passes these on to the mediator. The mediator shares these encryptions with *DS2* and *DS3*. All three service providers append the secret string to the values of their respective privacy-critical attributes such as ''Invoice'' and ''CustomerID''. For instance, Invoice *"489434"*, becomes *"489434Secret"* if $SecretString_{DS1-DS2-DS3} = Secret$. Next, each service provider calculates the SHA-256 (hashing algorithm) values of their respective privacy-critical attributes appended with the secret string (e.g., *"489434Secret"* for the example invoice value). Each service provider maintains a sorted table in the memory comprising the privacy critical attributes and the hash values of their respective appended forms. in Figure 1 *DS1* maintains a table *T(InvoiceHash, Invoice)*, *DS2* maintains two such tables as *T(InvoiceHash, Invoice)* and *T(CustomerIDHash, CustomerID)*, where ''*InvoiceHash*'' is created using $SecretString_{DS1-DS2-DS3}$ and ''*CustomerIDHash*'' is created using $SecretString_{DS2-DS4}$. The mediator executes a query on the databases with the hashed appended values as the input instead of the actual attribute values. For example, the mediator queries *DS2* with the ''*InvoiceHash*'' as input instead the ''Invoice'' as input.

F. MEDIATOR PICKS A SERVICE PROVIDER ACCORDING TO SERVICE COMPOSITION PLAN

To execute an input query, the mediator executes a number of sub-queries on individual data service providers based on the service composition plan (Steps 12, 13, and 14 of Algorithm 1). For example, *in Figure 1 a sub-query is executed on DS1 with "description" as input. Next, a sub-query is executed on DS2 by using the output of DS1 i.e.,* $Hash(Invoice + SecretString_{DS1-DS2-DS3})$. *A sub-query is also executed on DS3 using the same output of DS1. Finally, a sub-query is executed on DS4 using the output of DS2 as input,* $Hash(CustomerID + SecretString_{DS2-DS4})$.

G. HASHED VALUE GENERALIZATION USING K-ANONYMITY

Value generalization is the process wherein the mediator queries a generalized (at least K values) set of values from a data service provider, instead of just querying one specific value of interest x (Steps 18 and 19 of Algorithm 1, Steps 16 and 17 of Algorithm 2). In other words, if a query is to be executed using an input value x on a Data Service Provider *DSj*, then generalization results in the query being executed using *K-1* values in addition to the query with x. All the K possible values belong to *DSj* and K is called the maximum protection factor required by the parents of *DSj* in the service composition plan (Figure 1). The mediator computes the generalization of a value with the help of *DSj* in such a manner that *DSj* is unable to know which of the K values is the value x for which the query needed to be really executed. This ensures privacy to the extent that *DSj* can only guess (with a probability of $1/K$ of being correct) the values of x.

Why Hashed Value Generalization? Assuming two service providers, *DSi* and *DSj*, where *DSi* is the parent of *DSj*. The mediator shares the service composition plan with all the participating service providers including *DSi* and *DSj*. If the mediator executes a query on *DSj* using an output from *DSi* (*DSj*'s parent), then *DSj* comes to know that *DSi* has information on this value. This is possible because both *DSi* and *DSj* have the hash for a common value. This infringes upon the requirement of privacy and needs to be avoided. Hence, *hash value generalization* is used that results in *DSj* only being able to make a guess on the value of input data, with a probability of *1/K* of being correct. After a query is executed on a service provider with K, the mediator discards the *K-1* extraneous results and holds on to just the one actual result, by *filtering the false positives*.

Selectivity (Se) is an important factor that needs to be defined before a detailed description of Hashed Value Generalization is provided. Selectivity represents the number of output values from an in-memory table of a service provider over a given range. *Se(DSi, R)* is the number of output values in the range R of the in-memory table of *DSi*. For example, *the value of Se(DSi,* $[-\infty, +\infty]$*) is total number of elements in the table whereas Se(DSi, [319e5, 6e093]) is total number of elements between "319e5" and "6e093"*.

Hashed Value Generalization: For a given in-memory table of a child data service provider, the mediator follows the procedure below for hashed value generalization.

1) Mediator queries the selectivity of a service provider *DSi* with respect to the wide range of privacy critical values i.e Se(DSi, $[-\infty, +\infty]$) (i.e the whole table) along with the middle value (say *mid*) in the range of $]-\infty, +\infty[$.

2) If the selectivity value returned is no less than *2*K*, and the *mid* value is lexicographically more than *x*, then step 1 is repeated with the new range $]-\infty, mid]$.
3) Else, if the returned selectivity value is no less than *2*K*, and the *mid* value is lexicographically less than or equal to *x*, then step 1 is repeated with the new range $[mid, +\infty[$.
4) Else, if the returned selectivity value is less than *2*K*, then the current range comprises the generalized values of *x*. Instead of just *x*, the service provider *DSi* is queried with all the values in the range.
5) The mediator removes every result except that of *x*. This is called the removal of false-positives.

We have largely adopted this method from - Barhamgi *et al.* [13]. A limitation with their implementation is that it is susceptible to a privacy breach whereas we overcome it using an in-memory table and hashed values (discussed in III-G).

Privacy Analysis of Hashed Value Generalization: According to [13] the only privacy issue with the method is that the result of generalization changes with a change in the table. If the mediator does value generalization with a value (say *x*) twice from the same table with the data in the table being different the two times, then this results in value generalization with two different sets of data. However, one value remains common in both the results and that is the actual value being protected. This would enable the child service provider to guess that the common value is the actual value. The solution to this problem is that the result of value generalization should not change even when there is change in the table. To realise this, we use 'in-memory' tables to perform value generalization. The in-memory table remains unchanged for a given data service composition. Even when there is a change in the original table, the in-memory table remains unchanged for that data service composition. A new in-memory table is used for the next composition. Further, a child in a service composition plan shares with its parent the value of the number of queries executed on it by the mediator. This number, in an ideal scenario, is equal to K and represents the instructions given by the parent service to the mediator on the number of queries to be executed. If the value shared by the child is different from K, it indicates improper adherence to the K-Anonymity mechanism.

H. HOW AUTHENTICATION IS ACHIEVED?

Hashed Value Generalization, as discussed in Section III-G, only protects the value of a parent data service provider from the child data service provider in a service-composition plan. Authentication between two service providers is actually achieved using a secret string shared between the parent-child pairs (discussed in Section III-C). Owing to the shared secret string between parent-child pairs, the mediator is not be able to introduce a service provider from beyond the agreed upon service-composition plan. If, in a service composition plan, service provider *DSi* is the parent of service provider *DSj*, then the following steps provision security between *DSi* and *DSj* and this security is denoted by *Sij*.

1) *DSi* picks a secret string and shares it with *DSj* by encrypting it using $PRI_{DSi}(PUB_{DSj}(SecretString))$.
2) Both *DSi* and *DSj* concatenate the secret string with the privacy critical attributes.
3) Both *DSi* and *DSj* compute the hash of the concatenation in the above step.
4) *DSj* shares the number of queries executed on it by the mediator with *DSi*.

Whenever the mediator executes a query on *DSi*, the same returns the result of the query to the mediator after securing it using *Sij*. By doing this, *DSi* and *DSj* authenticate each other in the following way:

1) The parent *DSi* gets confirmation that no one can use its output other than *DSj*.
2) *DSi* also gets to know whether the mediator is following the value of K or not.
3) *DSj* also gets confirmation that the input for which the value is being generalized using K-Anonymity, is coming from *DSi*. This is because all the hashed values in *DSj*'s in-memory table, on which value generalization is be performed, are secured by *Sij*.

I. HOW IS EVERYTHING WORKING WHILE MAINTAINING PRIVACY?

The operations of a mediator and the service providers in a service-composition plan are described in detail in Algorithms 1 and 2 respectively. The following example describes the communication between these components done whilst maintaining security in Figure 1. In Figure 1 the security between *DS1*, *DS2* and *DS3* is denoted by *S123* for convenience of illustration. The three services share a common secret string because *DS2* and *DS3* are children of *DS1*. A query is executed on *DS1* with "Description" as the input. The result of *DS1* is Table T1["Description", "S123(Invoice)", "Quantity"]. It is important to note that the mediator cannot see the value of "Invoice". The mediator performs value generalization (using K-Anonymity) of "S123(Invoice)" on both *DS2* and *DS3* based on the value received from *DS1*. It then queries *DS2* using the generalized values of "S123(Invoice)" as input. *DS2* creates table T2["S123(Invoice)", "S24(CustomerID)"] for all the generalized values. Similarly, the mediator queries *DS3* using the generalized values of "*S*123(*Invoice*)" as input. *DS3* creates table T3["S123(Invoice)", "Invoice-Date"] for all the generalized values. The mediator then performs value generalization (using K-Anonymity) of "S24(CustomerID)" on *DS4* based on the value received from *DS2*. Next the mediator queries *DS4* using the generalized values of "S24(CustomerID)" as input. *DS4* creates table T4["S24(CustomerID)", "Country"]. In all the cases, the mediator eliminates the respective *K-1* data used for generalization and only retains the actual data. Finally, the mediator joins T1, T2, T3, and T4 to get the final result of the query. A simplified sequence diagram of the proposed method having only one parent and one child is shown in the Figure 3.

Algorithm 1 Mediator's Steps of Execution

Input: Query
Result: result_of_Query
1 **var** *ServiceProviders[]* stores reference to all the service providers;
2 **var** *query[]* stores sub-queries of the original query;
3 Create Service Composition Plan;
4 Store reference of all the service providers in service composition plan in *ServiceProviders[]*;
5 **for** Service_Provider s : ServiceProviders[] **do**
6 Share Service Composition Plan with s;
7 share certificate of s with everyone else in the ServiceProviders[];
8 **if** s is the first service provider in the composition **then**
9 **for** Service_Provider s1 : ServiceProviders[] **do**
10 Ask $PRI_S(PUB_{S1}(\text{“}secret''))$ from s and give it to S1;
11 **end**
12 **end**
13 Do query rewriting (i.e. divide the Query into sub-queries and store them in the array query[]);
14 **for** query q : query[] **do**
15 Find a service provider in ServiceProviders[] to execute q (Say *SP*);
16 **if** q does not have privacy constrained data from anyone else in ServiceProviders[] **then**
17 execute q;
18 **else**
19 Ask the value of K from the source of privacy constrained data of the query;
20 Do value generalization of privacy constrained data on *SP*;
21 Help *SP* to get confirmation from its parent (i.e source of privacy constrained data) about the value of K. Take the secured results of value generalization from the *SP* ;
22 Execute q on *SP* using all the results of value generalization;
23 Collect the secured results of query execution;
24 Remove false positive;
25 Help the *SP* to share the number of query executed on it with source of privacy constrained data ;
26 **if** q does not execute fine **then**
27 abort;
28 Do join operation if required;
29 **end**
30 return *result_of _Query*;

Algorithm 2 Service Provider's Steps of Execution

1 We refer the current service provider as *SP*;
2 Collect the service-composition plan from the mediator;
3 Store references of all the service providers (and their certificates) of the service-composition plan ;
4 Decide the value of K (its level of protection) for the value generalization using K-Anonymity and share it with the mediator;
5 **if** Mediator asks for a secret to be shared with others in composition plan **then**
6 Select a secret string say ''Secret'';
7 **for** All the service providers S in the service composition **do**
8 Share $PRI_{SP}(PUB_S(\text{“}Secret''))$ with mediator;
9 **end**
10 **if** $PRI_{SP\_child}(PUB_{SP}(valueofK))$ received **then**
11 **if** decided value of K matches with the received K **then**
12 send $PRI_{SP}(PUB_{SP\_child}(\text{“}OK''))$ to child via mediator;
13 **else**
14 send $PRI_{SP}(PUB_{SP\_child}(\text{“}Abort''))$ to child via mediator;
15 Collect $PRI_{SP}(PUB_{SP\_parent}(\text{“}Secret''))$ provided by the mediator (Line 9 Algorithm 1) and decrypt it;
16 Suffix the privacy critical attributes with the string ''ParentSecret'' and calculate $Hash(PrivacyCriticalValue + Secret)$;
17 Create a table in memory as: $T(PrivacyCriticalAttribute, Hash(PrivacyCriticalValue + Secret))$;
18 Sort the table created in step 6 with respect to $Hash(PrivacyCriticalValue + Secret)$;
19 Help the mediator for value generalization based on $Hash(PrivacyCriticalValue + Secret)$;
20 Cache the result of value generalization;
21 Send $PRI_{SP}(PUB_{SP_parent}(Count(\text{“}$result of value generalization''))))$ to parent via mediator;
22 Get confirmation from the parent;
23 **if** Confirmation from the parent is positive/ **then**
24 Give the secured result of query executed by mediator as $Hash(result + Secret)$ for all the attributes in result;
25 **else**
26 Abort the execution of query;

IV. EVALUATION

In this section, the proposed service composition method is evaluated on the criteria of security, privacy, and performance. The privacy and security aspects of the proposed method are first evaluated by assessing the behaviour of the system under standard likely 'threat' circumstances. This is followed by comparing the proposed method with existing state-of-the-art techniques in terms of various aspects of security and privacy. Finally, the proposed method is evaluated in terms of its performance wherein the execution time and resource requirements are assessed and shown to be within reasonable bounds.

A. DATASET

To evaluate the proposed method for privacy in data service composition, we downloaded an online retail dataset from the UCI Machine Learning Repository (link of the dataset). There is only one table in this dataset that contains 1,067,371 records. The schema of this table is as follows: *R(Invoice, StockCode, Description, Quantity, InvoiceDate, Price Customer, ID, Country)*. From this table, we created four different schemas: *R1(Invoice, Description, Quantity)*, *R2(Invoice, CustomerID)*, *R3(Invoice, InvoiceDate, Price)*, *R4(CustomerID, Country)*. R1 includes the description and quantity of an invoice, R2 includes the customer identification associated with the invoice, R3 includes the price and date of an invoice, and R4 has information on the country that a customer belongs to. We create four tables corresponding to the four schemas. We insert 65K, 130K, 195K, 260K, 325K, 390K, 455K and 520K records in each of the tables. That is, the first time that the query is executed 65 thousand records are inserted in each of the four tables, the next time 130 thousand records are inserted, and so on.

B. EXPERIMENTAL SETUP

We simulate a data service-composition scenario by providing the four schemas to four different data service providers to maintain, *DS1*, *DS2*, *DS3*, and *DS4*, correspondingly. The four services and a mediator are deployed as five standalone applications in Java on the same machine. The datasets are stored in a MySql server. All the processes described in Section III are implemented using appropriate Java APIs. For example, cryptography is implemented using packages ''java.security'', and ''java.crypto''. We run the experiments on an AMD Ryzen 7 3700U, 2.3GHZ processor running Windows 10 with 12GB RAM.

C. SECURITY AND PRIVACY ANALYSIS

This section first examines how the proposed method overcomes certain standard security challenges in data service composition. We use sequence diagrams to depict the normal flow (Figure 3) of a composition and to depict the various threats that a data service composition has to face (Figure 4). These threats comprise unethical behaviour of the mediator. The idea of using sequence diagrams for these representations is taken from the AVISPA tool [20]. For simplicity, a scenario of one mediator and two data service providers is assumed where one service provider is the parent and the other is the child.

The threats depicted in the sequence diagram are described below along with a brief discussion on how the proposed method overcomes them.

- ***Threat:*** *The mediator attempts to insert an alien service provider into the composition*
  Resolution: The mediator is unable to do this because the latter is not aware of the common secret key shared between the services and therefore cannot share this with the alien service. The only way that the alien service can gain access to the secret key is by sharing its certificate with one of the members of the service composition.
- ***Threat:*** *The mediator attempts to surreptitiously change the service-composition plan by changes the relationships between the participating service providers, from Parent → Child → Alien to Parent → Alien → Child*
  Resolution: The legitimate Child of Parent, waits for a message (i.e $PUB_{Child}(PRI_{Parent}(K))$) from the Parent before supporting the mediator in value generalization. As the value is encrypted using Child's public key it does not make sense to Alien. Therefore the mediator is unable to change the service composition in this manner.
- ***Threat:*** *This is another scenario wherein the mediator attempts to surreptitiously change the service-composition plan in the following manner: here also, Alien is considered a legitimate member of the composition plan and the correct plan is Parent → Child → Alien. This is modified to Alien → Child by the mediator and this modification happens after the Child has received a message from the Parent and needs to respond. The Alien replaces Parent and becomes the new parent of the child.*
  Resolution: In this case, the Child still considers the Parent as its parent. The Child sends a message (i.e $PUB_{parent}(PRI_{child}($ No. of iteration in value generalization$))$) to the Parent after assisting the mediator with value generalization. As the value is encrypted using the Parent's public key it does not make sense to the Alien. Therefore the mediator is unable to change the service composition in this manner.

The vulnerabilities and threats discussed above are by no means comprehensive and represent a small sample of possible threats. They are included to demonstrate the working of the proposed method.

Apart from overcoming the threats described, the proposed method ensures complete privacy between communicating service providers. No message exchanged between these service providers gets leaked. The proposed method ensures this privacy through value generalization by the mediator. The only thing that the mediator is trusted with is the encrypted intermediate value of messages exchanged between service providers in the composition.

1) COMPARISON WITH EXISTING TECHNIQUES

Let us now compare our work, in terms security and privacy, against other related methods in terms of the functionalities they provide. The work proposed by Yau and Yin [8] on data service composition is quite similar to our method, with the exception of two main differences.

- Yau and Yin [8] use a unique random factor is between each parent-child pairs which is analogous to the Secret in the proposed method.
- The major difference between our method and [8] is that the latter does not do value generalization and it is not able to maintain privacy between two service providers.

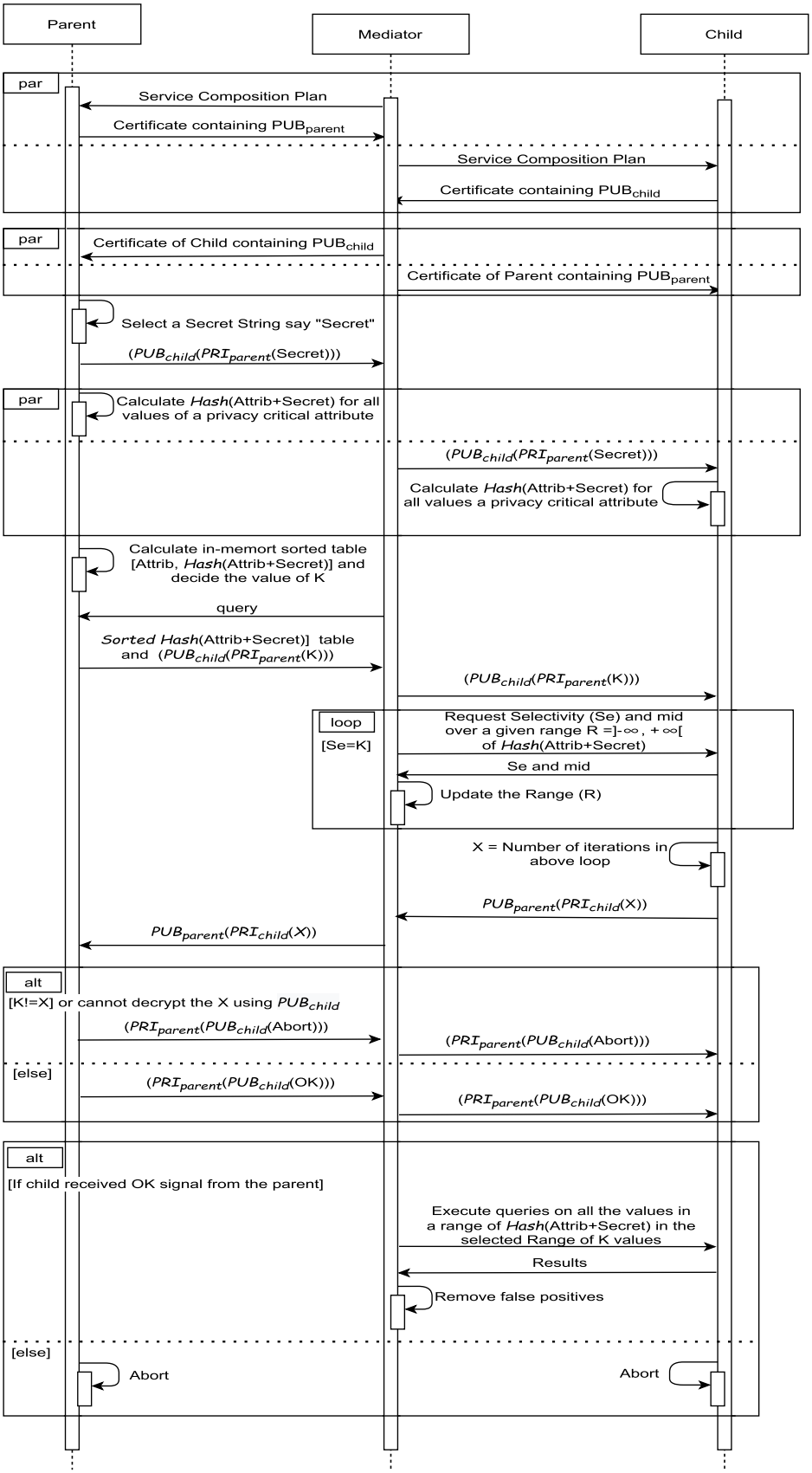

FIGURE 3. Sequence diagram of proposed data service composition with only two members.

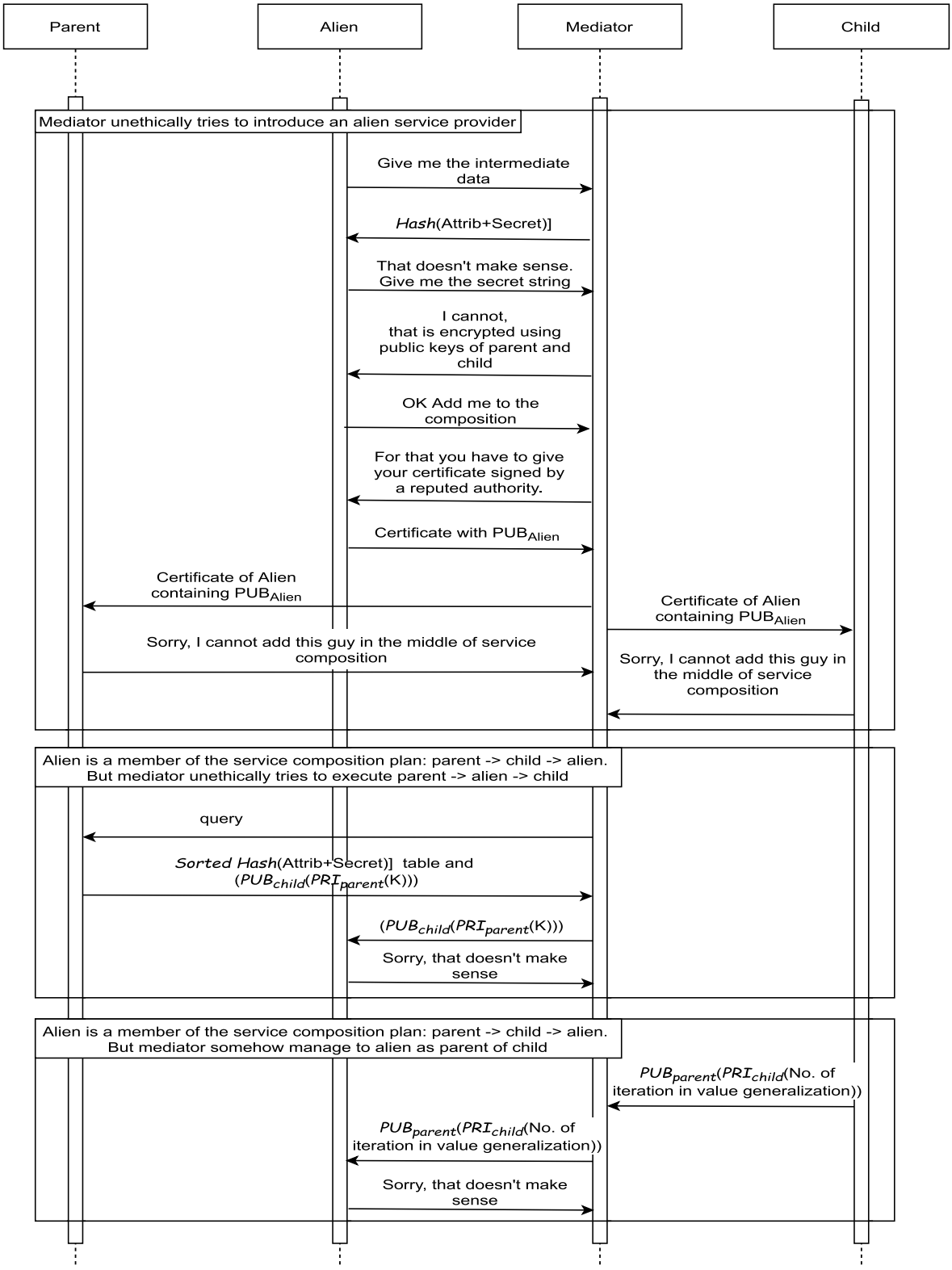

FIGURE 4. Security analysis of the proposed method having unethical mediator.

Tbahriti *et al.* [3] propose an approach that employs formal models for ensuring privacy. When two services lack compatibility in ensuring privacy, the approach prescribes that the mediator negotiate and make amends. This approach is quite different from ours in that there is no real cryptography involved.

The method proposed by Barhamgi *et al.* [13] is the latest in the field of data service composition.

- The technique uses OPES [14] encryption instead of ''Hashing with a common secret'', to hide intermediate data from the mediator. OPES encrypts data in a manner that two encrypted values can be compared without the need to be decrypted. The limitation of this technique, however, is that it cannot be used for encrypting non-numeric data.
- Just like our method, this technique also ensures privacy between two service providers through value generalization.
- The absence of the system of a common secret in this technique makes it vulnerable to the illegitimate inclusion of an alien service into a shared composition plan by an untrustworthy mediator.
- Also, the absence of authentication messages between parent and child exposes the service-composition plan to illegitimate alteration by an mediator.

Table 1 summarises the endeavours of various approaches at ensuring security and privacy in data service composition.

D. PERFORMANCE ANALYSIS

To analyse the performance of the proposed method, we recorded the execution time of a query with and without the applying the proposed method on the experimental data service composition set-up described earlier in this section. The query executed is: *how many lunchboxes were purchased by people of France on December 3, 2012?*. The execution of this query requires the involvement of all four service providers discussed in Section I. The service-composition plan to execute this query is shown in Figure 1. In the performance analysis results reported, the *process time* graphs show the actual CPU time expended to complete the execution of a query, whereas the *elapsed time* graphs shows the total time interval between the start and finish of query execution.

Figures 5, and 6 show that irrespective of whether it is the elapsed time or the process time, the required execution time increases with the number of queries executed. As the value of K is large, a large number queries is executed and the time of execution is long. Further, while performing value generalization, the number records in the database plays an important role. The more the number of records there are in the database, the more time that value generalization takes for a given value of K. Therefore, for a given value of K, the execution time increases as the number of records increases. On the other hand if no privacy is incorporated, the execution time remains constant.

Execution time also depends on the number of privacy critical data used in service composition. A large number of privacy critical data results in more value generalization and hence a large number of queries that need to be executed. In Figure 1, DS1 has a privacy critical attribute ''Invoice''. Figures 7 and 8 show that a large number of outputs from DS1 results in longer execution time for a given value of K. The Execution time of the proposed method is better than

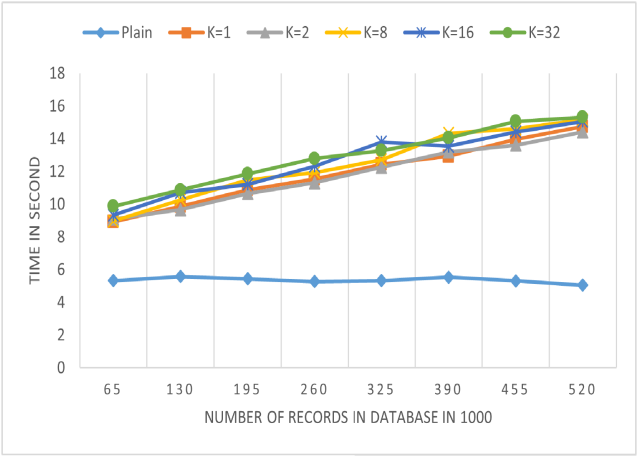

FIGURE 5. Process time of execution vs. number of records (DS1 has 9 records in output).

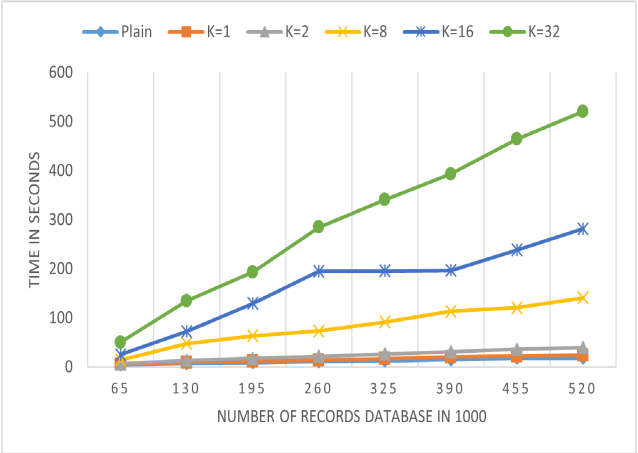

FIGURE 6. Elapsed time of execution vs. number of records (DS1 has 9 records in output).

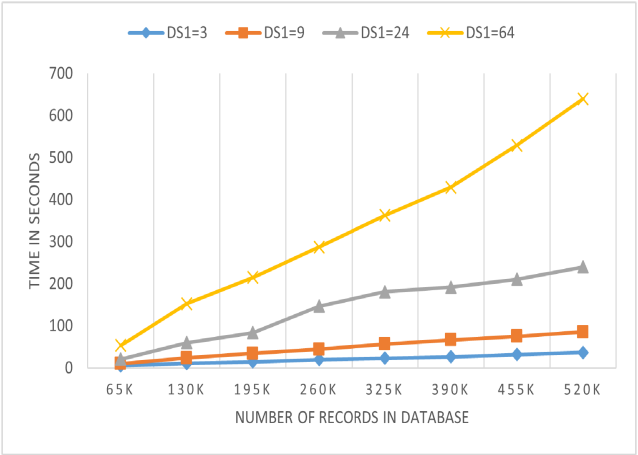

FIGURE 7. Elapsed time of execution vs. number of records (K = 4).

Barhamgi *et al.* [13]. Where to execute a query the method proposed by [13] takes more than an hour in its most secure form, the same proposed method takes hardly few minutes.

Therefore, the time required to execute a query depends on the following: 1) the number of actual execution

TABLE 1. Privacy features available with various methods in a data service-composition environment.

	Stephen S. Yau et.al. (2008)	SE Tbahriti et.al (2014)	M Barhamgi et.al (2019)	Proposed Method
Intermediate data is Visible to the mediator	No	Yes	No	No
Privacy between a parent-child pair in a service composition is preserved	No	No	Yes	Yes
Value generalization Used	No	No	Yes	Yes
Encryption of the Intermediate data	Yes	No	Yes	Yes
Common secret shared among participating services	yes	No	No	Yes
Supported type of Intermediate data	Any	Any	Integer	Any
Authentication between Service providers	Yes	No	No	Yes
Source service has assurance about destination of its output data	Yes	No	No	Yes
Source service has assurance about the proper use of K-Anonymity	No	No	No	Yes
Possibility of Replay attack by mediator	Yes	Yes	Yes	No
Trust on mediator	Average	Very High	Average	Low
Mutual trust among services	High	Very High	Average	Low

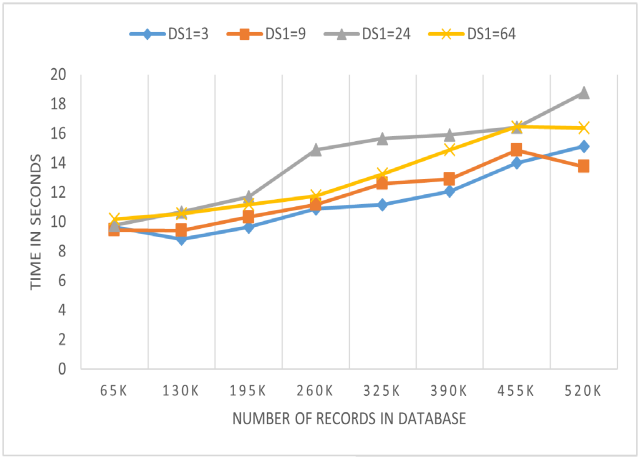

FIGURE 8. Process time of execution vs. number of records (K = 4).

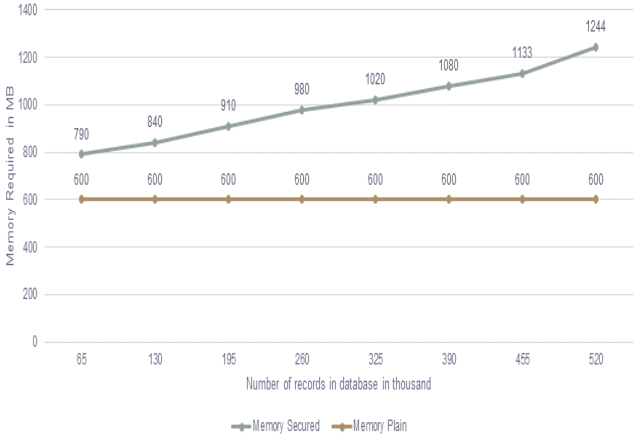

FIGURE 9. Memory consumption for query execution (K = 4).

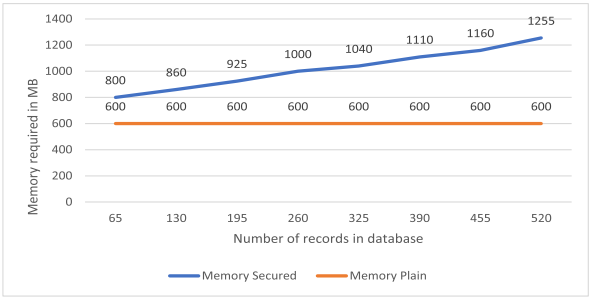

FIGURE 10. Memory consumption for query execution (K = 16).

instances, which is K times the number of privacy critical attributes; 2) the total number of service providers in the service-composition plan; 3) the total number of records in the database of each service provider, as the mediator performs a binary search on it for value generalization; and 4) the cost of the authentication process, which is a constant since it involves a fixed number of messages. If the number of privacy-critical attributes is P, a number of service providers in the service-composition plan is S, the average number of records per service provider is R, and K is the value of K in value generalization, then the time complexity of query execution is O(P*K*S + log(R)). This is the time complexity of the whole system of data service composition. The time taken to sort the in-memory table is not counted in the time complexity, as it is a separate one-time activity of each service provider. Each service provider also has to maintain a sorted table in memory, which contains all the records of that service provider. So each service provider has a space complexity of O(R).

The entire service composition set up is installed on the same Windows 10 system as discussed earlier. By executing the service composition, we are able to assess the total memory requirements. As is evident from Figures 9 and 10, if the value is not generalized, the memory requirements do not change with increasing number of records. With value generalization, however, an in-memory table needs to be maintained and therefore the memory requirements increase with increasing records.

V. CONCLUSION

This paper proposes a method for data service composition that maintains privacy between data service providers. The method assumes an untrustworthy mediator and, relying on the *k-anonymity* algorithm for value generalization and basic authentication and encryption, it provides for effective privacy in data service composition, mitigating the risks of the mediator sharing participant data with external entities and violating the predefined and agreed upon service-composition plan. Our method meets the key privacy requirements of sensitive-data providers participating in data

service compositions and its feasibility has been demonstrated through experimental evaluation with real-world data services.

REFERENCES

[1] D. Abadi, R. Agrawal, A. Ailamaki, M. Balazinska, P. A. Bernstein, M. J. Carey, S. Chaudhuri, J. Dean, A. Doan, M. J. Franklin, and J. Gehrke, "The Beckman report on database research," *Commun. ACM*, vol. 59, no. 2, pp. 92–99, Jan. 2016.

[2] R. Ranjan, O. Rana, S. Nepal, M. Yousif, P. James, Z. Wen, S. Barr, P. Watson, P. P. Jayaraman, D. Georgakopoulos, M. Villari, M. Fazio, S. Garg, R. Buyya, L. Wang, A. Y. Zomaya, and S. Dustdar, "The next grand challenges: Integrating the Internet of Things and data science," *IEEE Cloud Comput.*, vol. 5, no. 3, pp. 12–26, May 2018.

[3] S.-E. Tbahriti, C. Ghedira, B. Medjahed, and M. Mrissa, "Privacy-enhanced Web service composition," *IEEE Trans. Services Comput.*, vol. 7, no. 2, pp. 210–222, Apr. 2014.

[4] L. Sweeney, "K-anonymity: A model for protecting privacy," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, Oct. 2002.

[5] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "L-diversity: Privacy beyond k-anonymity," *Trans. Knowl. Discovery Data*, vol. 1, no. 1, p. 3, Mar. 2007.

[6] N. Li, T. Li, and S. Venkatasubramanian, "T-closeness: Privacy beyond k-anonymity and l-diversity," in *Proc. IEEE 23rd Int. Conf. Data Eng.*, Apr. 2007, pp. 106–115.

[7] M. Barhamgi, A. K. Bandara, Y. Yu, K. Belhajjame, and B. Nuseibeh, "Protecting privacy in the cloud: Current practices, future directions," *Computer*, vol. 49, no. 2, pp. 68–72, Feb. 2016.

[8] S. S. Yau and Y. Yin, "A privacy preserving repository for data integration across data sharing services," *IEEE Trans. Services Comput.*, vol. 1, no. 3, pp. 130–140, Jul. 2008.

[9] B. C. M. Fung, T. Trojer, P. C. K. Hung, L. Xiong, K. Al-Hussaeni, and R. Dssouli, "Service-oriented architecture for high-dimensional private data mashup," *IEEE Trans. Services Comput.*, vol. 5, no. 3, pp. 373–386, Feb. 2012.

[10] N. Mohammed, X. Jiang, R. Chen, B. C. M. Fung, and L. Ohno-Machado, "Privacy-preserving heterogeneous health data sharing," *J. Amer. Med. Inform. Assoc.*, vol. 20, no. 3, pp. 462–469, May 2013.

[11] S. Wang, D. Agrawal, and A. El Abbadi, "Towards practical private processing of database queries over public data," *Distrib. Parallel Databases*, vol. 32, no. 1, pp. 65–89, Mar. 2014.

[12] K. Nayak, X. S. Wang, S. Ioannidis, U. Weinsberg, N. Taft, and E. Shi, "GraphSC: Parallel secure computation made easy," in *Proc. IEEE Symp. Secur. Privacy*, May 2015, pp. 377–394.

[13] M. Barhamgi, C. Perera, C.-M. Yu, D. Benslimane, D. Camacho, and C. Bonnet, "Privacy in data service composition," *IEEE Trans. Services Comput.*, vol. 13, no. 4, pp. 639–652, Jul. 2020.

[14] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order preserving encryption for numeric data," in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, 2004, pp. 563–574.

[15] R. Sion and B. Carbunar, "On the practicality of private information retrieval," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, Feb. 2007, pp. 1–10.

[16] M. Barhamgi, D. Benslimane, and B. Medjahed, "A query rewriting approach for Web service composition," *IEEE Trans. Services Comput.*, vol. 3, no. 3, pp. 206–222, Jul. 2010.

[17] M. Barhamgi, D. Benslimane, Y. Amghar, N. Cuppens-Boulahia, and F. Cuppens, "PrivComp: A privacy-aware data service composition system," in *Proc. 16th Int. Conf. Extending Database Technol. (EDBT)*, 2013, pp. 757–760.

[18] U. Srivastava, K. Munagala, J. Widom, and R. Motwani, *Query Optimization Over Web Services*. Citeseer, 2006.

[19] R. Housley, W. Ford, W. Polk, and D. Solo, *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*, document RFC 2459, Jan. 1999.

[20] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuéllar, P. H. Drielsma, P. C. Héam, O. Kouchnarenko, J. Mantovani, and S. Mödersheim, "The AVISPA tool for the automated validation of Internet security protocols and applications," in *Proc. Int. Conf. Comput. Aided Verification*, Berlin, Germany: Springer, Jul. 2005, pp. 281–285.

GYAN PRAKASH TIWARY received the B.Tech. degree in information technology from the Birsa Institute of Technology Sindri (B.I.T. Sindri), Dhanbad, India, in 2010, and the M.Tech. degree in computer applications from the Indian Institute of Technology (Indian School of Mines) Dhanbad, Dhanbad, India, in 2013. He is currently pursuing the Ph.D. degree in computer science and engineering with the Indian Institute of Technology Indore, Indore, India.

From 2019 to 2020, he was awarded with the Overseas Visiting Doctoral Fellowship by the Science and Engineering Research Board, Government of India, to work as Visiting Research Scholar at the University of Alberta, Canada. His research interests include the service oriented architecture, privacy and security accepts of web services, document compression, and cryptography.

Mr. Tiwary received fellowship and grants, including the Overseas Visiting Doctoral Fellowship (Science and Engineering Research Board), the Visvesvaraya Ph.D. Scheme Fellowship during his Ph.D. studies at IIT Indore, and the GATE Fellowship during the study of M.Tech. degree at IIT (ISM) Dhanbad.

ELENI STROULIA (Member, IEEE) is currently a Professor with the Department of Computing Science, University of Alberta. From 2011 to 2016, she held the NSERC/AITF Industrial Research Chair on service systems management with IBM. Her research interests include addressing industry-driven problems and adopting AI and machine-learning methods to improve or automate tasks. Her flagship project in the area of health care is the Smart Condo in which she investigates the use of technology to support people with chronic conditions live independently longer and to educate health-science students to provide better care for these clients. In 2011, the Smart-Condo Team received the UofA Teaching Unit Award. She has played leadership roles with the GRAND and AGE-WELL NCEs. She has supervised more than 60 graduate students and PDFs, who have gone forward to stellar academic and industrial careers. In 2018, she received the McCalla Professorship. In 2019, she was recognized with the Killam Award for Excellence in Mentoring. Since January 2020, she has been the Director of the AI4Society Signature Area.

ABHISHEK SRIVASTAVA received the Ph.D. degree from the University of Alberta, Canada, in 2011. He is currently an Associate Professor with the Discipline of Computer Science and Engineering, Indian Institute of Technology Indore. His group at IIT Indore has been involved in research on service-oriented systems most commonly realized through web-services. Recently, the group has been interested in applying these ideas in the realm of the Internet of Things. The ideas explored include coming up with technology agnostic solutions for seamlessly linking heterogeneous the IoT deployments across domains. Further, the group is also delving into utilizing machine learning adapted for constrained environments to effectively make sense of the huge amounts of data that emanate from the vast network of the IoT deployments.