Филипп Б. Буртыка
Южный федеральный университет, ул. Большая Садовая, 105/42, г. Ростов-на-Дону, 344006, Россия, e-mail: bbfilipp@ya.ru, ORCID iD 0000-0002-7879-802X

МЕТОДЫ ОРГАНИЗАЦИИ ПРОИЗВОЛЬНЫХ КОНФИДЕНЦИАЛЬНЫХ ЗАПРОСОВ К КРИПТОГРАФИЧЕСКОЙ ОБЛАЧНОЙ БАЗЕ ДАННЫХ С ПОМОЩЬЮ ПОЛНОСТЬЮ ГОМОМОРФНОГО ШИФРОВАНИЯ

DOI: http://dx.doi.org/10.26583/bit.2017.2.03

Аннотация. Обработка запросов к зашифрованной базе данных без ее расшифрования в последнее время активно исследуется как специалистами по криптографии, так и исследователями в области БД. Такую обработку позволяют проводить различные виды т.н. обрабатываемого шифрования (ОШ), а также специальные архитектуры систем управления базами данных (СУБД), использующие эти виды шифрования. Самыми известными видами ОШ являются шифрование, сохраняющее порядок, гомоморфное шифрование, функциональное шифрование, поисковое шифрование и шифрование, сохраняющее свойства. На основе этих видов шифрований строятся различные СУБД, наиболее известными примерами которых могут служить CryptDB, Monomi, Arx и СУБД исследователей из Новосибирска. Эти СУБД строятся на применении комбинации различных видов ОШ, например, шифрования, сохраняющего порядок, гомоморфного шифрования и традиционных видов шифрования. Однако, такой подход может вызвать проблемы с конфиденциальностью. Наилучшим подходом с точки зрения защищенности было бы построение криптографической СУБД с использованием только гомоморфного шифрования. Препятствием к этому пока что является недостаточная эффективность самого гомоморфного шифрования, а также неполное решение комплекса вопросов, связанных с необходимостью конфиденциального принятия решения в недоверенной среде. В данной работе предлагаются методы для решения этих вопросов, в частности для организации выполнения произвольного конфиденциального запроса к зашифрованной реляционной БД с помощью ПГШ. Также предлагается модель запроса к БД, разделяющая условие запроса на атомарные предикаты и связывающее условие. Один из предлагаемых методов направлен на сохранение конфиденциальности связывающего условия, другие на сохранение конфиденциальности атомарных предикатов. Параметры предложенных методов позволяют реализовать их с использованием уже существующих схем гомоморфного шифрования. Предлагаемые методы могут служить основой для построения систем защищенных криптографических облачных баз данных.

Ключевые слова: конфиденциальные запросы, криптографическая база данных, полностью гомоморфное шифрование, защищенные вычисления, секретные вычисления, недоверенная среда, криптографический протокол, конфиденциальное получение информации

Для цитирования. **БУРТЫКА, Филипп Б.** Методы организации произвольных конфиденциальных запросов к криптографической облачной базе данных с помощью полностью гомоморфного шифрования. Безопасность информационных технологий, [S.l.], v. 24, n. 2, p. 24-34, June 2017. ISSN 2074-7136. Доступно на: <https://bit.mephi.ru/index.php/bit/article/view/102>. Дата доступа: 13 Dec. 2017. doi:http://dx.doi.org/10.26583/bit.2017.2.03.

Filipp B. Burtyka
Southern federal university, Bolshaya Sadovaya St., 105/42 Rostov-on-Don, 344006, Russia, e-mail: bbfilipp@ya.ru, ORCID iD 0000-0002-7879-802X

The Techniques for Arbitrary Secure Querinq to Encrypted Cloud Database Using Fully Homomorphic Encryption

DOI: http://dx.doi.org/10.26583/bit.2017.2.01

Abstract. The processing of queries to an encrypted database without its decryption has been actively investigated recently by both cryptographers and database researchers. Such a work is allowed by various types of so-called Processable encryption (PE), as well as special architectures of database management systems (DBMS) which use these types of encryption. The most known types of PEs are order-preserving encryption, homomorphic encryption, functional encryption, searchable encryption, and property-preserving encryption. Based on these types of encryption the various DBMSs are built, the most famous of which are CryptDB, Monomi, Arx and DBMS by researchers from Novosibirsk. These DBMSs are built on the basis of various types of PEs, for example order-preserving encryption, homomorphic encryption and traditional block encryption. However this approach can cause privacy problems. The best approach from the security viewpoint is to build a cryptographic database using only homomorphic encryption. An obstacle to this is insufficient efficiency of the existing homomorphic encryption schemes and incomplete solution of a set of issues related to ensuring the confidentiality of decisions making in an untrusted environment. In this paper we propose the techniques for solving these problems, in particular for organization of execution arbitrary secure query to the encrypted relational database using fully homomorphic encryption. Also we propose a model of query condition that splits query into atomic predicates and linking condition. One of reposed technique is aimed at ensuring the security of linking condition of queries, others keep security of atomic predicates. The parameters of the proposed techniques make it possible to implement them using the already existing homomorphic encryption schemes. The proposed techniques can be a basis for building secure cryptographic cloud databases.

Keywords: Private Queries, Encrypted Database, Fully Homomorphic Encryption, Secure Computations, Secret Computations, Untrusted Environment, Cryptographic Protocol, Private Information Retrieval, Querying Encrypted Data

For citation. **BURTYKA, Filipp B.** The Techniques for Arbitrary Secure Quering to Encrypted Cloud Database Using Fully Homomorphic Encryption. IT Security (Russia), [S.l.], v. 24, n. 2, p. 24-34, June 2017. ISSN 2074-7136. Available at: <https://bit.mephi.ru/index.php/bit/article/view/102>. Date accessed: 13 Dec. 2017. doi:http://dx.doi.org/10.26583/bit.2017.2.03.

На сегодняшний день облачные вычисления представляют собой ситуацию недоверенной вычислительной среды: нарушитель безопасности может быть как в канале передачи данных, так и на стороне сервера – за 2013 год организацией “Online Trust Alliance” (“Сообщество доверия в Сетú”) зафиксировало около 740 млн. инцидентов безопасности [1], ещё большее число осталось незамеченным, и год от года это количество неуклонно возрастает. Единственным противодействием в этой ситуации является полное шифрование данных без расшифрования их вне клиентского компьютера. Вместе с тем естественным является желание использовать облачные ресурсы не только как хранилище данных, но и как вычислительные мощности. Таким образом, необходимо уметь проводить все необходимые вычисления над зашифрованными данными.

Инструментом для этого является т.н. обрабатываемое шифрование (ОШ, англ. processable encryption, PE) которое позволяет проводить над зашифрованными данными те или иные манипуляции без расшифрования. К такому типу шифрования относятся: гомоморфное шифрование (ГШ), шифрование сохраняющее порядок (ШСП, англ. order-preserving encryption, OPE), функциональное шифрование (ФШ, англ. functional encryption), поисковое шифрование (ПШ), шифрование, сохраняющее свойства (ШСС) и ряд других. Рассмотрим подробнее каждый из этих видов шифрования.

Суть ГШ состоит в том, что на множестве открытых текстов рассматривается одна или более операции [2], каждой из которых соответствует операция на множестве шифртекстов так, что при проведении операций над шифртекстами после расшифрования результат получается идентичным тому, как если бы соответствующие операции проводились бы над открытыми данными. Частным случаем гомоморфного шифрования является полностью

гомоморфное шифрование (ПГШ), которое гомоморфно по набору операций над открытыми данными, позволяющему выразить любую арифметическую или логическую функцию и, следовательно, теоретически провести любые вычисления над зашифрованными данными без их расшифрования [3]. Механизм действия ШСП [4] состоит в том, что на множестве шифртекстов вводится отношение порядка, «изоморфное» отношению порядка на множестве открытых текстов. ФШ характеризуется тем, что [5] позволяет вычислять некоторые заданные специальным ключом функции от зашифрованных данных, при этом результат вычислений может быть представлен в открытом виде. ПШ [6] позволяет зашифровать набор текстовых документов так, что для этого зашифрованного набора возможно проводить поиск документов, содержащих заданное зашифрованное ключевое слово.

Перейдем теперь к рассмотрению того, как эти виды шифрования используются для защиты облачных вычислений. Одними из самых распространенных типов вычислений в облаках являются алгоритмы работы с реляционными базами данных (см., например, сервис Amazon RDS [7]), такие как условные запросы (например, SQL-запросы). Выполнение такого запроса может, например, заключаться в том, что из БД выбираются записи, удовлетворяющие некоторым критериям, например, у которых заданный атрибут лежит в фиксированном диапазоне или точно равен какому-то значению. Запрос такого вида называется неагрегированным, что означает, что длина ответа на этот запрос не фиксирована (т.е. может содержать произвольное количество значений, зависящее от условий). Помимо неагрегированных запросов к БД, должна быть возможность проводить и агрегированные вычислительные запросы, такие как вычисление среднего или минимального/максимального значения по некоторому атрибуту A_i, в которых объем возвращаемых данных не зависит от содержимого БД.

Модель условия запроса

Ключевым моментом защиты облачной СУБД является обеспечение конфиденциальности запроса. В данном разделе предлагается модель декомпозиции условия запроса к БД и анализ защищенности существующих криптографических решений с использованием этой модели.

Пусть $\mathcal{R}(A_1, \dots, A_\tau))$ – реляционная схема [8] степени τ, которой соответствует база данных $\mathcal{D}$. К базе данных $\mathcal{D}$, состоящей из набора кортежей $(\alpha_1, \dots, \alpha_n)$, возможны запросы на добавление нового кортежа, либо запросы на извлечение кортежей. В последнем случае используется оператор $\mathcal{Q}$, извлекающий из $\mathcal{D}$ все кортежи $(\alpha_{i_1}, \dots, \alpha_{i_{n'}})$, удовлетворяющие некоторому условию на атрибуты A_i. На SQL-подобном языке можно записать такой запрос как

SELECT * FROM $\mathcal{R}$ WHERE condition (1)

Условие condition может быть записано в виде

$$\mathcal{B}(\mathcal{P}_1(A_1), \dots, \mathcal{P}_\tau(A_\tau)), \quad (2)$$

где $\mathcal{P}_i$ – *атомарные предикаты* для атрибутов A_i, а $\mathcal{B}$ – *связывающее условие*, булева формула от τ переменных. Условие вида (2) встречается и в агрегированных запросах вида

SELECT COUNT(*) FROM $\mathcal{R}$ WHERE condition (3)

Теперь рассмотрим ситуацию, когда БД находится в зашифрованном виде на недоверенном сервере, т.е. сервер хранит набор зашифрованных кортежей $(\overline{\alpha}_1, \dots, \overline{\alpha}_n)$ (здесь и далее по тексту будем обозначать как $\overline{m}$ зашифрованное значение элемента данных m, при этом предполагается, что сложные структуры данных шифруются поразрядно), и к ней необходимо *произвести запросы типа (1),(3) не производя расшифрования ни на каком этапе в процессе получения ответа* (отметим, что запросы на добавление записей в зашифрованную БД не вызывают проблем с конфиденциальностью, вопрос же удаления

записей выходит за рамки рассмотрения данной работы). Дополнительно можно ставить задачу скрытия параметров запроса Q т.е. атомарных предикатов $\mathcal{P}_i$ и/или связывающего условия $\mathcal{B}$ или, другими словами, невозможности узнавания этих параметров без ключа расшифрования.

Существует много работ, посвященных решению задач организации обработки зашифрованной реляционной БД в недоверенной среде без расшифрования с использованием различных видов ОШ.

Условно эти работы можно разделить на две группы. К первой относятся работы практической направленности, т.е. имеющие программную реализацию, работающую за приемлемое время для БД промышленных размеров (более 2^{30} записей). Сюда можно отнести CryptDB [9], MONOMI [10], программный комплекс, разработанный исследователями из Новосибирска [11]. Во всех этих работах предлагается следующее решение: разные атрибуты шифровать разными шифрами, при этом выбор шифра, применяемого к атрибуту A_i, определяется видом атомарного предиката $\mathcal{P}_i$, который необходимо будет вычислять над данными этого атрибута. Для неагрегированных запросов вида (1) в [9-11] предполагают, что предикат $\mathcal{P}_i(A_i)$ является либо предикатом сравнения на равенство вида $A_i = a$, либо предикатом сравнения порядка вида $A_i > a, A_i < a$. Для работы с предикатами сравнения на равенство используются детерминированные блочные шифры (такие как DES, AES, ГОСТ), а для предикатов сравнения порядка используется ШСП. Если же необходимо обрабатывать агрегированные арифметические запросы (сумма, среднее, дисперсия и т.д.) для атрибута A_i, то к нему применяют гомоморфное шифрование. В случае если по одному атрибуту нужно уметь проводить запросы с разными видами предикатов, то значения этого атрибута дублируются, а дубли шифруются разными шифрами.

Отметим, что в случае использования детерминированного шифрования и ШСП, неагрегированные запросы к БД выполняются тривиально т.к. клиент зашифровывает сами данные a, но не шифрует ни атомарные предикаты, ни связывающее условие $\mathcal{B}$. В этом случае, хотя сервер не знает данных, лежащих в БД, и значения a, он получает результат вычисления предикатов $\mathcal{P}_i(A_i)$ в незашифрованном виде и использует его для принятия решения о соответствии данной записи параметрам запроса. Ясно, что в этом случае при проведении запроса недоверенная сторона получит некоторую информацию, поскольку узнает, сколько записей удовлетворяют запросу клиента.

Данная общая концепция в указанных работах реализуется немного по-разному. В частности, в CryptDB [9] используется сочетание трёх видов шифрования: блочного AES, аддитивно гомоморфного шифрования Пэйе и варианта ШСП, предложенного авторами [9], которое было ноу-хау проекта, поскольку, по словам разработчиков, достигает максимума возможной защищенности для такого типа криптосистем. В работе же [11] используются только шифры, разработанные самими авторами.

В целом, такой подход, однако, ведет к проблеме с безопасностью, поскольку в этих условиях становятся возможны некоторые специфические, но довольно эффективные виды атак [12], [13]. Еще одним недостатком является то, что ШСП было признано неспособным обеспечить необходимый уровень конфиденциальности в принципе [3].

Работы второй категории имеют лишь теоретическое значение, хотя иногда и предлагают некоторые экспериментальные «proof of concept» реализации. В работах [14] и [15] были предложены способы организации работы с зашифрованными ячейками памяти с произвольным доступом (в нашем контексте то же самое что БД) при выполнении алгоритмов с зависимостями по данным (что необходимо для принятия решения по зашифрованному условию) с использованием так называемого многоаргументного функционального шифрования (МФШ). Однако это решение обладает рядом недостатков: в первую очередь, необходимо иметь действующий экземпляр такой криптосистемы

(которого на текущий момент нет), однако даже в случае достаточно высокой криптостойкости самого МФШ существуют угрозы от злонамеренных функциональностей, т.е. возможность узнать с помощью специального ключа критические сведения о данных, позволяющие раскрыть их (дешифровать). Далее, подготовительная работа, производимая клиентом перед запуском вычислений в недоверенной среде по сложности сопоставима с самим вычислением, таким образом, может дать выигрыш производительности для клиента только в очень специфических случаях (например, экспоненциальное количество запусков одной и той же функциональности над разными наборами данных).

Большую перспективу имеют решения с использованием ПГШ, однако параметры существующих полностью гомоморфных криптосистем сдерживают практическое применение этих решений. Было предпринято много попыток преодоления трудностей связанных с системами защиты БД, основанных на использовании ПГШ, и созданию и оптимизации методов и формул, реализующих эффективно реализующих необходимую для БД функциональность.

Одной из первых работ на эту тему была [16], в которой анализировалось как использовать ПГШ для организации запросов к БД, выходящих за пределы агрегатных функций и численных вычислений, таких как выборка, запросы по диапазону и соединенные. Было показано, что с использованием полностью гомоморфного шифрования, которое позволяет складывать и умножать биты открытого текста в зашифрованном виде возможно осуществлять сложные неагрегированные запросы в недоверенной среде без расшифрования, такие как выборка по диапазону значений, соединение таблиц, группировки. В этом случае на одном из этапов данные гомоморфно шифруются, а ответ возвращается клиенту в т.н. буфере результатов. При этом для неагрегированных запросов не гарантированно, что все результаты смогут быть восстановлены из буфера, хотя это может быть сделано с большой вероятностью. В числе информации, открытой для недоверенной стороны остается вычисляемый предикат, используемый в качестве условия, а также верхняя оценка на количество данных соответствующих неагрегированным запросам.

В работе [17] были предложены похожие решения. Однако протокол для неагрегированных запросов из [17] имеет некоторые особенности. На первом этапе сервер вычисляет гомоморфно булеву формулу $\mathcal{B}\big(\mathcal{P}_1(A_1), \dots, \mathcal{P}_1(A_\tau)\big)$ для каждой строки таблицы. Результатом является $\overline{1}$ или $\overline{0}$. Затем сервер гомоморфно складывает полученные значения и отправляет клиенту зашифрованное число $\overline{n}$ записей таблицы, соответствующих запросу. На втором этапе сервер получает от клиента в открытом виде верхнюю оценку n' на число n ($n' \geq n$) и возвращает (после вычислений и гомоморфной сортировки) n' записей, удовлетворяющих запросу. Такой протокол дает гарантию, что все результаты будут получены клиентом, однако имеет уязвимости, позволяющие проводить некоторые специфические атаки.

В работе [18] был рассмотрен вопрос представления алгоритмов обработки запроса над зашифрованными с помощью ПГШ данными в виде набора схем из функциональных элементов (СФЭ). В случае ПГШ над битами это будут обычные комбинационные схемы. В [18] описаны базовые варианты СФЭ для различных видов запросов и приведены их оптимизации. При этом оптимизации проводились заточенные под использование так называемого пакетированного варианта ПГШ, позволяющего проводить вычисления параллельно над несколькими наборами открытых данных. Рассматривались конъюнктивные, дизъюнктивные и пороговые запросы. Тип запроса не скрывался. В качестве предикатов рассматривались только сравнения на равенство. Для неагрегированных запросов в [18] сервер отправлял клиенту всю зашифрованную таблицу целиком. В этой таблице значения атрибутов записей, не соответствующих запросу превращались в ноль посредством умножения всех значений записи на результат

гомоморфного применения к этой записи булевой формулы (2). Т.е коммуникационная сложность была эквивалентна отсылке всей БД и оптимизация подобная [16] и [17] не производилась.

В работе [19] были более подробно проанализированы схемные сложности предложенных СФЭ, приведены результаты экспериментов, а также был предложен интересный протокол для обработки неагрегированных запросов. Этот протокол включал интерполяцию полинома, подобно протоколу из [20], что позволило избежать вычислительно трудоемкой гомоморфной сортировки. Данное решение предполагает (как и в [16]) априорное знание клиентом количества записей в БД, соответствующих запросу. С этой целью был предложен следующий механизм: для каждого типа запроса и каждого набора параметров, который интересует клиента, он заводит у себя ячейку в которой хранит число – верхнюю оценку количества записей, подходящее под этот запрос с указанными параметрами. Время от времени клиент посылает на сервер запросы на вычисление в зашифрованном виде этой оценки и сохраняет в ячейку максимальное значение. В случае необходимости произвести запрос с данным набором параметров клиент извлекает из ячейки это значение верхней оценки и отправляет его на сервер в составе запроса. Недостатком данной системы является то, что клиенту необходимо запоминать для каждого запроса значение оценки, что может потребовать больших ресурсов памяти в случае, если клиенту требуется вычислять достаточно разнообразные запросы.

В статье [21] рассматриваются дополнительные оптимизации СФЭ для агрегированных и неагрегированных запросов из [18], в частности используются два метода оптимизации – первый позволяет реализовать в пакетном режиме единичные базовые операции, а второй позволяет оптимизировать СФЭ для целочисленного сложения даже для случая меньших чем число пространств открытых текстов. Для СФЭ проверки равенства были предложены оптимизации, уменьшающие мультипликативную глубину.

В работе [22] также рассматривалась задача организации запросов по диапазону с использованием ПГШ. Была предложена реализация с использованием ПГШ из [23], и проведены эксперименты для разных значений параметров уровня криптостойкости.

Авторами [24] был предложен и проанализирован фреймворк для динамической интерпретации и компиляции запросов в булевы СФЭ. Также был рассмотрен вопрос автоматической оптимизации булевых СФЭ, получаемых в процессе преобразования программного кода и проведены эксперименты для оценки эффективности фреймворка.

В работе [25] были предложены протоколы для конфиденциального вычисления трех видов запросов конъюнктивного, дизъюнктивного и порогового, причем впервые предлагалось зашифровывать от недоверенного сервера то, с каким именно из этих трех типов запросов обратился клиент.

Однако ни в какой из имеющихся работ не была рассмотрена задача организации *произвольного* конфиденциального запроса к зашифрованной гомоморфной криптосистемой базе данных и задача полного зашифрования всех параметров запроса.

Метод обеспечения конфиденциальности произвольного запроса к зашифрованной базе данных

Как уже было сказано, впервые возможность скрытия с помощью шифрования типа запроса к полностью гомоморфно зашифрованной БД обсуждалась в работе [25], однако авторы ограничились возможностью делать один из трех типов запросов – пороговый, конъюнктивный и дизъюнктивный (т.е. в нашей терминологии речь идет об ограничении на связывающее условие – оно может быть только конъюнкцией всех переменных, дизъюнкцией всех переменных, либо пороговым условием, принимающем истинное значение тогда и только тогда когда количество истинных переменных превышает

пороговое значение). Однако, что если мы хотим делать к зашифрованной реляционной БД запросы достаточно произвольного вида и использовать шифрование для сохранение конфиденциальности вида запроса? В данном разделе предлагается метод для использования произвольного связывающего условия.

Любая булева функция от τ переменных может быть выражена в виде полинома Жегалкина (булева многочлена) [26]:

$$\sum_{i=0}^{2^\tau} \beta_i \cdot x_0^{i_0} \cdot \ldots \cdot x_{\tau-1}^{i_{\tau-1}}, \tag{4}$$

где $i_0, \ldots, i_{\tau-1} \in \{0,1\}$ – разряды битового представления числа i (i_0 – самый младший разряд), а β_i – коэффициенты, изменяя которые можно получить всевозможные функции. В зависимости от коэффициентов его можно превратить в любой полином от τ переменных.

Рассмотрим пример для трех переменных x_0, x_1, x_2. Для трех переменных формула (4) примет вид

$$\beta_0 \oplus \beta_1 x_0 \oplus \beta_2 x_1 \oplus \beta_3 x_2 \oplus \beta_4 x_0 x_1 \oplus \beta_5 x_1 x_2 \oplus \beta_6 x_0 x_2 \oplus \beta_7 x_0 x_1 x_2. \tag{5}$$

Пороговый запрос для трех переменных, срабатывающий при двух и более переменных выражается полиномом Жегалкина

$$x_0 x_2 \oplus x_1 x_2 \oplus x_0 x_1 ,$$

что соответствует формуле (5) при наборе коэффициентов $\beta_0 = \beta_1 = \beta_2 = \beta_3 = \beta_7 = 0$, $\beta_4 = \beta_5 = \beta_6 = 1$. Пороговый запрос для трех переменных, срабатывающий при одной и более переменных совпадает с дизъюнкцией трех переменных и имеет вид

$$x_0 \oplus x_1 \oplus x_2 \oplus x_0 x_1 \oplus x_1 x_2 \oplus x_0 x_2 \oplus x_0 x_1 x_2,$$

который может быть получен из (5) при наборе коэффициентов $\beta_0 = 0,\ \beta_1 = \beta_2 = \beta_3 = \beta_4 = \beta_5 = \beta_6 = \beta_7$. Пороговый запрос, срабатывающий только при трех переменных, совпадает с конъюнкцией трех переменных и выглядит как

$$x_0 x_1 x_2 ,$$

что соответствует формуле (5) с набором коэффициентов $\beta_0 = \beta_1 = \beta_2 = \beta_3 = \beta_4 = \beta_5 = \beta_6 = 0$ и $\beta_7 = 1$.

Основной идеей предлагаемого метода является то, что в случае, когда коэффициенты β_i зашифрованы, невозможно определить, какая именно функция вычисляется. Таким образом, подставляя вместо переменных x_i зашифрованные результаты вычисления предикатов и гомоморфно вычисляя формулу (4) с использованием зашифрованных коэффициентов $\overline{\beta}_i$ недоверенный вычислитель получает зашифрованный бит, обозначающий выполнимость условия запроса для данной зашифрованной записи.

Оценим мультипликативную глубину [27] схемы вычисления полинома (4) а также общий объем вычислений. В формуле (4) моном наибольшей степени – это произведение τ переменных предикатов и переменной-коэффициента β_i, а, как известно [27], мультипликативная глубина СФЭ, вычисляющей произведение $\tau + 1$ переменных, будет $\lceil \log_2(\tau + 1) \rceil$. Следовательно, это и будет мультипликативной глубиной схемы.

Поскольку полином (4) является функцией симметрической по переменным $x_0, \ldots, x_{\tau-1}$, то можно воспользоваться построением из [27] добавив на выход схемы лишь домножения на коэффициенты β_i и в итоге получив общую мультипликативную сложность СФЭ порядка $\tau + 3\sqrt{\tau} + 2^\tau$.

Коммуникационная же сложность метода составляет $O(2^\tau)$, поскольку в ходе выполнения запроса клиент должен переслать серверу 2^τ зашифрованных коэффициентов $\overline{\beta}_i$. Таким образом, предлагаемый метод можно легко применять при количестве атрибутов в БД до 25-30.

Обеспечение конфиденциальности типа предиката и его параметров

Отдельно рассмотрим скрытие типа предиката и его параметров. В общем случае, поскольку речь идет о конкретной компьютерной системе, каждый из атрибутов A_i может принимать лишь конечное число значений из своей области определения $\mathrm{dom}\, A_i$. При этом предикат $\mathcal{P}_i$ задает некоторое конечное подмножество в $\mathrm{dom}\, A_i$, а размер этого подмножества изменяется от одного элемента до $|\mathrm{dom}\, A_i|$ ($|\mathrm{dom}\, A_i|$ обозначает размер множества $\mathrm{dom}\, A_i$).

Частный случай, когда предикат определялся на числовом множестве и являлся сравнением на равенство и больше/меньше рассматривался в работах [28],[29]. При этом константы, с которыми сравнивалось проверяемое в предикате значение, задавались в зашифрованном виде. Заметим, что не только числовую, но и любую другую конечную область можно представить в виде числового множества, отождествив ее с числами от 1 до $|\mathrm{dom}\, A_i|$. Например, условие, определяющее длину текстовой строки, может быть выражено как диапазон значений строк. Таким образом, если подмножество, задаваемое предикатом $\mathcal{P}_i$, представляет собой непрерывный интервал на отрезке $[1, |\mathrm{dom}\, A_i|]$, то его можно задать с помощью двух констант, при этом для определения принадлежности этому интервалу использовать алгоритм компаратора [28], который, принимая на вход два целочисленных значения одинаковой разрядности, выдает истину, если первое значение больше второго. Алгоритм компаратора для чисел заданной разрядности выражается в виде полинома Жегалкина и может быть выполнен над шифртекстами (при этом результат работы будет тоже в зашифрованном виде) [29].

В случае же если предикат должен задавать подмножество, состоящее из небольшого количества значений, лежащих далеко друг от друга, то можно воспользоваться полиномиальным представлением множества, как в [20]. При этом степень полинома определяет размер задаваемого множества элементов, которые будут корнями этого полинома. Зашифровывая его коэффициенты, тем самым делаем конфиденциальными параметры предиката.

Вышесказанное можно обобщить, составив формулу для вычисления атомарного предиката в зашифрованном виде

$$\overline{\mathcal{P}}_i = \left(\overline{\gamma}_i \oplus \overline{R}_i \oplus \overline{\gamma}_i \overline{R}_i\right)\left(\overline{\zeta_i} \oplus \overline{\delta}_i \oplus \overline{P}_i \oplus \overline{\delta}_i \overline{P}_i\right), \tag{6}$$

где $\overline{R_i}$ – зашифрованный результат вычисления диапазонного компаратора, $\overline{\gamma}_i$ – зашифрованный бит активации диапазонного компаратора, $\overline{P}_i$ – зашифрованный результат вычисления полиномиального условия, $\overline{\delta}_i$ – зашифрованный бит активации полиномиального условия, $\overline{\zeta_i}$ – зашифрованный бит инверсии полиномиального условия. Зашифрованный результат вычисления полиномиального условия равен $\overline{1}$ тогда и только тогда, когда значение представляющего множество полинома в вычисляемом значении предиката равно нулю. Зашифрованные значения $\overline{\gamma}_i, \overline{\delta}_i, \overline{\zeta}_i$ присылаются клиентом вместе с другими параметрами условия запроса.

Заключение

Представлена модель условия запроса к БД, позволяющая эффективно декомпозировать его на атомарные предикаты и связывающее условие. Предложены методы организации зашифрования атомарных предикатов и связывающего условия запроса. Новизна методов по сравнению с известными методами обеспечения конфиденциальности запросов к криптографической БД, использующими полностью гомоморфное шифрование, заключается в том, что они дают возможность выразить любое условие на атрибуты, при этом фактически не раскрывая никакой информации об этом условии в недоверенную среду. Реализация и применение данных методов впервые делают возможным организовать произвольные конфиденциальные запросы к зашифрованной облачной базе данных

эффективно в достаточной мере для практического использования и вместе с тем гарантированную защищенность исключительно криптостойкостью используемой полностью гомоморфной криптосистемы. Представленные методы могут быть реализованы с использованием любой полностью гомоморфной криптосистемы.

Более полное и математически строгое изложение будет приведено в отдельной статье.

СПИСОК ЛИТЕРАТУРЫ:

1. “Online trust alliance finds data breaches spiked to record levelin 2013”. URL: https://otalliance.org/news-events/press-releases/online-trust-alliance-finds-data-breaches-spiked-record-level-2013 (дата обращения: 25.01.2017)
2. Варновский Н. П., Шокуров А. В. Гомоморфное шифрование // Труды ИСП РАН. 2007. Вып. 12. С.27-36.
3. Бабенко Л. К., Буртыка Ф. Б., Макаревич О. Б., Трепачева А. В. Полностью гомоморфное шифрование (обзор) // Вопросы защиты информации. 2016. № 3. С. 3-25.
4. Boldyreva A., Chenette N., O’Neill A. Order-preserving encryption revisited: Improved security analysis and alternative solutions //Advances in Cryptology – CRYPTO 2011. С. 578-595.
5. Boneh D., Sahai A., Waters B. Functional encryption: Definitions and challenges // Proceedings of Theory of Cryptography Conference. – Springer Berlin Heidelberg, 2011. С. 253-273.
6. Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R. Searchable symmetric encryption: improved definitions and efficient constructions //Journal of Computer Security. – 2011. – Т. 19. – №. 5. С. 895-934.
7. Amazon Relational Database Service (Amazon RDS). URL: https://aws.amazon.com/rds/ (дата обращения: 25.01.2017)
8. Дейт К. Дж. Введение в системы баз данных. М.: Вильямс, 2001. 1072 с. ISBN 5-8459-0138-3.
9. Popa, R. A., Redfield, C., Zeldovich, N., Balakrishnan, H. CryptDB: protecting confidentiality with encrypted query processing //Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles. 2011. С. 85-100. DOI: 10.1145/2043556.2043566
10. Tu, S., Kaashoek, M. F., Madden, S., Zeldovich, N. Processing analytical queries over encrypted data //Proceedings of the VLDB Endowment. – VLDB Endowment, 2013. – Т. 6. – №. 5. С. 289-300.
11. Shatilov K., Boiko V., Krendelev S., Anisutina D., Sumaneev A. Solution for secure private data storage in a cloud // Federated Conference on Computer Science and Information Systems (FedCSIS), IEEE, 2014, С. 885-889. DOI: 10.15439/2014F43
12. Бабенко Л.К., Трепачева А.В. Анализ защищенности одной криптографической системы для защиты конфиденциальности данных в облачных вычислениях // Безопасность информационных технологий. 2016. № 1. С. 11-15
13. Naveed M., Kamara S., Wright C. V. Inference attacks on property-preserving encrypted databases //Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015. С. 644-655.
14. Goldwasser, S., Kalai, Y., Popa, R. A., Vaikuntanathan, V., Zeldovich, N. Reusable garbled circuits and succinct functional encryption //Proceedings of the forty-fifth annual ACM symposium on Theory of computing. 2013. С. 555-564. DOI: 10.1145/2488608.2488678
15. Goldwasser, S., Gordon, S. D., Goyal, V., Jain, A., Katz, J., Liu, F. H., Zhou, H. S. Multi-input functional encryption //Advances in Cryptology–EUROCRYPT 2014. 2014. С. 578-602.
16. Wang S., Agrawal D., El Abbadi A. Is Homomorphic Encryption the Holy Grail for Database Queries on Encrypted Data? // University of California, Santa Barbara, USA. 2012. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.228.4576&rep=rep1&type=pdf (дата обращения: 26.01.2017)
17. Mani M., Shah K., Gunda M. Enabling secure database as a service using fully homomorphic encryption: Challenges and opportunities //arXiv preprint 1302.2654, 2013. URL: https://arxiv.org/pdf/1302.2654 (дата обращения: 24.01.2017)
18. Cheon J. H., Kim M., Kim M. Search-and-compute on encrypted data //Proceedings of International Conference on Financial Cryptography and Data Security, 2015. С. 142-159.
19. Kim, M., Lee, H. T., Ling, S., Wang, H. On the Efficiency of FHE-based Private Queries //IEEE Transactions on Dependable and Secure Computing. – 2016.
20. Boneh, D., Gentry, C., Halevi, S., Wang, F., Wu, D. J. Private database queries using somewhat homomorphic encryption //Proceedings of International Conference on Applied Cryptography and Network Security, 2013. С. 102-118.

21. Cheon J. H., Kim M., Kim M. Optimized search-and-compute circuits and their application to query evaluation on encrypted data //IEEE Transactions on Information Forensics and Security. 2016. Т. 11, №. 1. С. 188-199. DOI: 10.1109/TIFS.2015.2483486
22. Wei S. Somewhat homomorphic encryption scheme for secure range query process in a cloud environment: диссертация на соискание степени магистра компьютерных наук. Laurentian University of Sudbury, 2015.
23. Van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V. Fully homomorphic encryption over the integers //Proceedings of Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2010. С. 24-43.
24. Stan O., Carpov S., Sirdey R. Dynamic Execution of Secure Queries over Homomorphic Encrypted Databases //Proceedings of the 4th ACM International Workshop on Security in Cloud Computing. – ACM, 2016. – С. 51-58.
25. Kim, M., Lee, H. T., Ling, S., Ren, S. Q., Tan, B. H. M., Wang, H. Better Security for Queries on Encrypted Databases. // IACR Cryptology ePrint Archive. 2016. № 470. URL: http://eprint.iacr.org/2016/470.pdf (дата обращения: 25.01.2017)
26. Марченков С. С, Замкнутые классы булевых функций. Москва: Физматлит. 2000. 128 с.
27 Boyar J., Peralta R., Pochuev D. On the multiplicative complexity of boolean functions over the basis (∧,⊕, 1) //Theoretical Computer Science. – 2000. – Т. 235. – №. 1. – С. 43-57.
28. Togan M. A FHE-based evaluation for searching on encrypted data // International Conference on Communications (COMM), IEEE, 2016. С. 291-296.
29. Togan M., Morogan L., Plesca C. Comparison-based applications for fully homomorphic encrypted data // Proceedings of the Romanian Academy – series A: Mathematics, Physics, Technical Sciences, Information Science. 2015. Т. 16, С. 329-338.

REFERENCES:

[1]. “Online trust alliance finds data breaches spiked to record levelin 2013”. Available at: https://otalliance.org/news-events/press-releases/online-trust-alliance-finds-data-breaches-spiked-record-level-2013 (accessed 25.01.2017)
[2]. N.P. Varnovskij, A.V. Shokurov. Homomorphic encryption. Trudy ISP RAN [The Proceedings of ISP RAS], 2007. Vol. 12, pp. 27-36. (in Russian).
[3]. Babenko L. K., Burtyka F. B., Makarevich O. B. Trepacheva A.V. Fully Homomorphic Encryption (A Survey). Voprosy zashhity informacii [The information defence issues]. 2015, no. 3, pp. 3–25. (in Russian)
[4]. Boldyreva A., Chenette N., O’Neill A. Order-preserving encryption revisited: Improved security analysis and alternative solutions. Advances in Cryptology – CRYPTO 2011. pp. 578-595.
[5]. Boneh D., Sahai A., Waters B. Functional encryption: Definitions and challenges. Proceedings of Theory of Cryptography Conference. – Springer Berlin Heidelberg, 2011. pp. 253-273.
[6]. Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R. Searchable symmetric encryption: improved definitions and efficient constructions.; Journal of Computer Security. 2011. Vol. 19. №. 5. pp. 895-934.
[7]. Amazon Relational Database Service (Amazon RDS). Available at: https://aws.amazon.com/rds/ (accessed on 25.01.2017)
[8]. Date C. J. An introduction to database systems. Moscow, Williams Publ., 2001, 1072 p. (in Russian)
[9]. Popa, R. A., Redfield, C., Zeldovich, N., Balakrishnan, H. CryptDB: protecting confidentiality with encrypted query processing. Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles. 2011. pp. 85-100. DOI: 10.1145/2043556.2043566
[10]. Tu, S., Kaashoek, M. F., Madden, S., Zeldovich, N. Processing analytical queries over encrypted data. Proceedings of the VLDB Endowment. – VLDB Endowment, 2013. Vol. 6. №. 5. pp. 289-300.
[11]. Shatilov K., Boiko V., Krendelev S., Anisutina D., Sumaneev A. Solution for secure private data storage in a cloud. Federated Conference on Computer Science and Information Systems (FedCSIS), IEEE, 2014, pp. 885-889. DOI: 10.15439/2014F43
[12]. Babenko L. K., Trepacheva A.V. Security Analysis of One Solution for Secure Private Data Storage in a Cloud. Bezopasnost' informacionnyh tehnologij [IT Security (Russia)], 2016, no. 1, pp. 11-15 (in Russian).
[13]. Naveed M., Kamara S., Wright C. V. Inference attacks on property-preserving encrypted databases. Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015, pp. 644-655.
[14]. Goldwasser, S., Kalai, Y., Popa, R. A., Vaikuntanathan, V., Zeldovich, N. Reusable garbled circuits and succinct functional encryption. Proceedings of the forty-fifth annual ACM symposium on Theory of computing, 2013, pp. 555-564. DOI: 10.1145/2488608.2488678
[15]. Goldwasser, S., Gordon, S. D., Goyal, V., Jain, A., Katz, J., Liu, F. H., Zhou, H. S. Multi-input functional encryption. Advances in Cryptology – EUROCRYPT 2014, pp. 578-602.
[16]. Wang S., Agrawal D., El Abbadi A. Is Homomorphic Encryption the Holy Grail for Database Queries on Encrypted Data?; University of California, Santa Barbara, USA. 2012. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.228.4576&rep=rep1&type=pdf (accessed on: 26.01.2017).

[17]. Mani M., Shah K., Gunda M. Enabling secure database as a service using fully homomorphic encryption: Challenges and opportunities //arXiv preprint 1302.2654, 2013. Available at: https://arxiv.org/pdf/1302.2654 (accessed on: 24.01.2017).
[18]. Cheon J. H., Kim M., Kim M. Search-and-compute on encrypted data. Proceedings of International Conference on Financial Cryptography and Data Security, 2015. pp. 142-159.
[19]. Kim, M., Lee, H. T., Ling, S., Wang, H. On the Efficiency of FHE-based Private Queries. IEEE Transactions on Dependable and Secure Computing, 2016.
[20]. Boneh, D., Gentry, C., Halevi, S., Wang, F., Wu, D. J. Private database queries using somewhat homomorphic encryption. Proceedings of International Conference on Applied Cryptography and Network Security, 2013. pp. 102-118.
[21]. Cheon J. H., Kim M., Kim M. Optimized search-and-compute circuits and their application to query evaluation on encrypted data. IEEE Transactions on Information Forensics and Security, 2016, vol. 11, no. 1, pp. 188-199. DOI: 10.1109/TIFS.2015.2483486
[22]. Wei S. Somewhat homomorphic encryption scheme for secure range query process in a cloud environment: Master's thesis in Computer Science. Laurentian University of Sudbury, 2015.
[23]. Van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V. Fully homomorphic encryption over the integers. Proceedings of Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2010. pp. 24-43.
[24]. Stan O., Carpov S., Sirdey R. Dynamic Execution of Secure Queries over Homomorphic Encrypted Database. Proceedings of the 4th ACM International Workshop on Security in Cloud Computing. – ACM, 2016. – pp. 51-58.
[25]. Kim, M., Lee, H. T., Ling, S., Ren, S. Q., Tan, B. H. M., Wang, H. Better Security for Queries on Encrypted Databases. IACR Cryptology ePrint Archive, 2016, no. 470. Available at: http://eprint.iacr.org/2016/470.pdf (accessed on 25.01.2017)
[26]. Marchenkov S. S, Zamknutye klassy bulevyh funkcij. Moskva: Fizmatlit. 2000. Pp. 128.
[27]. Boyar J., Peralta R., Pochuev D. On the multiplicative complexity of boolean functions over the basis ($\wedge$,$\oplus$, 1). Theoretical Computer Science. – 2000. Vol. 235. №. 1. pp. 43-57.
[28]. Togan M. A FHE-based evaluation for searching on encrypted data. International Conference on Communications (COMM), IEEE, 2016, pp. 291-296.
[29]. Togan M., Morogan L., Plesca C. Comparison-based applications for fully homomorphic encrypted data. Proceedings of the Romanian Academy – series A: Mathematics, Physics, Technical Sciences, Information Science. 2015, vol. 16, pp. 329-338.

Поступила в редакцию – 21 февраля 2017 г. Окончательный вариант – 20 мая 2017 г.
Received – February 21, 2017. The final version – May 20, 2017.