Hacia la seguridad criptográfica en sistemas DaaS

Rafael Álvarez
Departamento de Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante
Email: ralvarez@dccia.ua.es

Juan Santonja
Departamento de Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante
Email: jsl2@alu.ua.es

Antonio Zamora
Departamento de Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante
Email: zamora@dccia.ua.es

***Resumen*—El nuevo paradigma de computación en la nube posibilita la prestación de servicios por terceros. Entre ellos, se encuentra el de las bases de datos como servicio (*DaaS*) que permite externalizar la gestión y alojamiento del sistema de gestión de base de datos.**

Si bien esto puede resultar muy beneficioso (reducción de costes, gestión simplificada, etc.), plantea algunas dificultades respecto a la funcionalidad, el rendimiento y, en especial, la seguridad de dichos servicios.

En este trabajo se describen algunas de las propuestas de seguridad en sistemas *DaaS* existentes y se realiza un análisis de sus características principales, introduciendo un nuevo enfoque basado en tecnologías no exclusivamente relacionales (*NoSQL*) que presenta ventajas respecto a la escalabilidad y el rendimiento.

***Palabras clave*—Base de datos (database), Cloud Computing, Cifrado homomórfico (Homorphic encryption).**

I. Introducción

En la actualidad estamos asistiendo al rápido despliegue del modelo de computación llamado *cloud computing* (computación en la nube). Este paradigma de computación ofrece un nuevo modelo de prestación de servicios tecnológicos y de negocio. A medida que va creciendo la oferta de proveedores, también lo hace la diversidad de servicios que ofrecen, así tenemos servicios *SaaS* (software as a service), *IaaS* (Infrastructure as a Service), *PaaS* (Platform as a Service) y *DaaS* (Database as a Service). En *DaaS*, el modelo permite ofrecer bases de datos como servicio.

Es este último caso el que resulta más interesante dada la problemática relacionada tanto con la seguridad como con la funcionalidad cuando externalizamos una base de datos. En concreto, los sistemas de bases de datos relacionales son los más utilizados para modelar problemas reales y administrar datos dinámicamente.

Los sistemas de gestión de bases de datos relacionales (*RDBMS*, Relational Database Management Systems) son uno de los elementos esenciales en los sistemas de computación actuales, ya que permiten almacenar y administrar datos en forma de tablas, de forma razonablemente sencilla mediante la definición de relaciones entre tuplas y elementos estructurales que limitan la existencia de duplicados, permitiendo la unión y búsqueda de elementos dentro de las tablas de forma óptima.

Las bases de datos relacionales utilizan el lenguaje de consulta estructurado o *SQL* (Structured Query Language). Dicho lenguaje, basado en el álgebra relacional, permite realizar tanto consultas de información a la base de datos como de modificación de estructura.

Son muchos los *RDBMS* comerciales existentes en la actualidad, entre los más extendidos se encuentran *MySQL* (y *MariaDB*), *PostgreSQL*, *Oracle*, *DB2*, *INFORMIX* y *Microsoft SQL Server*. Todos ellos, junto a las características propias del sistema de gestión de datos, poseen controles de seguridad como la definición de roles de acceso, logs de auditoría o, en los más avanzados, sistemas de cifrado de la información.

Curino et al. [4] indican que el uso de sistemas *DaaS* en el *cloud* es interesante desde el punto de vista económico por dos motivos, fundamentalmente: el primero, relacionado con la reducción de consumo energético ya que los costes son menores cuando los recursos son compartidos por varios usuarios; el segundo, relacionado con los costes de gestión, tanto de licencias como gastos administrativos que son menores en sistemas compartidos.

Independientemente de los motivos económicos, existen una serie de factores (véase [5]) que incentivan el uso de servicios *DaaS*: la escalabilidad horizontal que permite que los recursos puedan ser ampliados casi sin límite en el *cloud*; la velocidad de despliegue de aplicaciones e infraestructura que es mucho más rápida en sistemas compartidos que en sistemas propios; la flexibilidad en la contratación de servicios específicos evitando los costes de aquellos elementos que no son necesarios y, por último, la mayor fiabilidad de los proveedores de *cloud* que disponen de sistemas redundantes de respaldo e infraestructura, mejorando la disponibilidad de los servicios.

Aun cuando son muchas las ventajas, existen una serie de inconvenientes (véase [5]) que deben considerarse a la hora de elegir un sistema *DaaS*, como son la velocidad, el rendimiento, los costes asociados a la gestión de grandes volúmenes de información (*Big Data*) y la pérdida de control sobre la información. Es este último aspecto, el de la seguridad, el eje fundamental del presente trabajo.

Al externalizar un sistema relacional y ubicarlo en el *cloud* la información queda expuesta al administrador del sistema o cualquier elemento con acceso directo a la base de datos. Una posible solución para proteger los datos, garantizando la confidencialidad de los mismos, consiste en cifrar la información; con el inconveniente de que al realizar dicha acción, algunas de las propiedades básicas de los sistemas relacionales no son viables. El reto consiste, pues, en construir un esquema de

ISBN: 978-84-9717-323-0

cifrado de la base de datos que permita la viabilidad de dichas propiedades (véase [16]).

Entre esas propiedades básicas destacamos las siguientes operaciones:

1. Relaciones entre tablas: Las tablas se relacionan entre sí mediante claves.
   a) Clave Primaria: permite identificar un registro de forma única.
   b) Clave Ajena: aparece en una tabla haciendo referencia a la información de otra tabla.
2. Operación Insertar: Añadir registros a una tabla
3. Operación Consulta: Seleccionar registros dentro de una tabla
   a) Consulta de atributos alfanuméricos
   b) Consulta de intervalos
   c) Consulta de agregación
4. Operación Modificar: Actualización de datos de una tabla.
5. Operación Eliminar: Borrado de registros

Existen diversas soluciones a este problema propuestas en la literatura. En este trabajo se describen y analizan algunas de ellas y se propone una nueva alternativa basada el paradigma no exclusivamente relacional (*NoSQL*).

II. Sistemas DaaS relacionales

II-A. Sistemas con cifrado en descanso (at rest)

Muchos de los productos existentes basan su seguridad exclusivamente en el cifrado del almacenamiento, de esta forma los datos se descifran de forma transparente al ser accedidos y se cifran al ser guardados, permaneciendo en claro mientras perduren en memoria. A pesar de la eficiencia en el rendimiento, el principal inconveniente que plantea este modelo consiste en que el proveedor elige (y, por tanto, dispone de) las claves de cifrado.

Aunque este modelo viene motivado por exigencias normativas, no resulta adecuado en aquellos casos en los que el proveedor no deba tener acceso a los datos, como ocurre en arquitecturas de tipo *DaaS* con datos sensibles.

Este modelo es adecuado para la protección ante un eventual robo de medios físicos, como unidades de disco o cintas de seguridad, ya que impediría el acceso a la información en claro. Las operaciones que realiza el sistema relacional hacen uso de los datos en memoria, que se encuentran en claro.

Algunas de las soluciones más utilizadas para este modelo son *Oracle* [10] y *SQL Server* [9].

II-B. Esquema de cifrado homomórfico completo

Aun cuando el modelo no tiene una implementación práctica viable en la actualidad, la propuesta de Gentry [6] resolvería completamente el reto propuesto. Consiste en un sistema de cifrado homomórfico, capaz de soportar tanto las operaciones de suma como de producto.

El principal problema de esta propuesta radica, como se ha indicado, en que su implementación no es viable con los medios computacionales disponibles en la actualidad. Por ejemplo, el sistema homomórfico parcial, requiere para el sistema más pequeño (512 dimensiones) un ancho de palabra de 200.000 bits, lo cual muestra la magnitud del problema. La clave pública usada en el sistema totalmente homomórfico tiene un tamaño de 17 MB y necesita 2.4 segundos para generarse. El sistema mayor (32768 dimensiones) requiere dos horas para generar la clave y ocupa 2.3GB [2].

II-C. Propuesta de L.M.X. Rodríguez

El esquema de cifrado para bases de datos relacionales propuesto por Rodríguez (véase [16]) pretende conseguir dos objetivos: garantizar la confidencialidad de la información y permitir que las consultas puedan ser procesadas por un sistema de base de datos relacional convencional. La propuesta utiliza diferentes tipos de primitivas criptográficas: un cifrador por bloques (AES), un cifrador homomórfico (Paillier [11]) y un cifrador que preserva el orden (Boldyreva [3]). Cada uno permite solventar limitaciones distintas a la hora de realizar consultas relacionales sobre un sistema de bases de datos cifrado.

Cifrador por bloques (AES): dado que los cifradores por bloques en modo directo (*ECB*) son deterministas (obtienen siempre el mismo texto cifrado para el mismo texto en claro con igual clave), permiten realizar las consultas que involucren campos alfanuméricos tanto para búsqueda (*SELECT*) como condicionales (*WHERE*). En esta propuesta se ha seleccionado el algoritmo *AES* por su buen rendimiento y ser un estándar bien conocido.

Cifrador basado en homomorfismos (Paillier): en las bases de datos es habitual realizar operaciones de totales y subtotales. Para permitir que el servidor realice operaciones que impliquen sumas, en esta propuesta se emplea un criptosistema homomórfico bajo la suma, como el de Paillier [11], en el que el resultado del producto de dos textos cifrados es idéntico al cifrado de la suma de esos textos sin cifrar; de esta manera se pueden sumar los datos sin conocerlos en claro.

Cifrador que preserva el orden (Boldyreva): en este tipo de cifrador, el texto cifrado presenta la característica de preservar el orden numérico del texto en claro; posibilitando la realización de consultas que impliquen la evaluación de un intervalo de información, estableciendo límites tanto inferiores como superiores.

La propuesta de Rodríguez [16] consta de tres elementos esenciales: el proceso de cifrado, el modelo de almacenaje y el proceso de consulta de la información cifrada.

- El proceso de cifrado recibe como entrada un registro de texto en claro el cual es analizado para elegir el método de cifrado apropiado para cada campo: los campos numéricos se cifran simultáneamente mediante el cifrador basado en homomorfismos y el que preservan el orden mientras que los alfanuméricos se cifran con el cifrador por bloques.
- El modelo de almacenaje se ve condicionado por el proceso de cifrado, dado que por cada campo numérico en claro se generan dos campos cifrados. El acceso a

cada campo cifrado se realiza en función de la operación requerida en la consulta.

- El proceso de consulta de información cifrada requiere la traducción de dichas consultas. El cliente solicita la consulta en claro y se analiza para determinar qué tipo de algoritmo aplicar: para el caso de restricciones alfanuméricas se accede a los campos cifrados con AES, para las comparaciones numéricas se accede a los campos cifrados con Boldyreva y, finalmente, para los cálculos que impliquen sumas se accede a los campos cifrados con Paillier.

Las limitaciones identificadas en el desarrollo de esta propuesta incluyen la imposibilidad de la ejecución de determinadas consultas, como búsquedas textuales mediante expresiones regulares, multiplicaciones, divisiones, números en coma flotante y fechas.

II-D. CryptDB

En la propuesta de Popa, et al. (véase [12], [13], [14], [15]) se desarrolla un sistema para garantizar la confidencialidad de la información en base a la ejecución de consultas SQL sobre datos cifrados utilizando un conjunto de esquemas de cifrado eficientes.

Este sistema permite gestionar las claves de cifrado a nivel de usuario, de modo que en una tabla puede haber datos de varios usuarios y sólo el usuario autorizado es capaz de descifrar los datos que le pertenecen.

CryptDB está formado por dos elementos: el sistema de bases de datos y el servidor de aplicaciones; y tiene dos objetivos fundamentales: limitar el acceso a la información por parte del proveedor de la base de datos y mantener la confidencialidad en caso de que la seguridad de todo el sistema se viera comprometida.

Al igual que en la propuesta de Rodríguez, los campos son cifrados de distintas maneras en función del tipo de dato y la operación a realizar. Para ello, se define una estructura de capas: igualdad, orden, búsqueda y suma. Cada dato es cifrado tantas veces como sea necesario (mediante el algoritmo adecuado) en función de las operaciones que se puedan realizar sobre el mismo.

Los cifrados que se pueden encontrar en las capas son: aleatorio, determinístico, preservando el orden, homomórfico y para relaciones.

La arquitectura del sistema consta de dos partes: un proxy de base de datos y el sistema de gestión de bases de datos relacional sin modificar. El proxy se encarga de traducir las consultas al formato adecuado para su ejecución en el sistema de gestión de bases de datos cifrado. En dicho servidor, se almacena la estructura de las tablas para realizar el proceso de traducción consultas. El servidor de base de datos tiene una serie de funciones definidas por el usuario para llevar a cabo algunas de las rutinas de cifrado/descifrado.

Al ser un sistema funcional, los creadores del mismo lo han podido probar como motor de base de datos para aplicaciones habituales en internet, como PHPbb o HOTCPR. Los resultados obtenidos muestran niveles de rendimiento y ejecución de consultas muy aceptables. A pesar de ello, hay un conjunto de consultas básicas, relacionadas con ordenaciones y uniones, que no pueden ejecutarse.

Por otra parte el tamaño de la base de datos aumenta significativamente, debido a que un mismo campo necesita ser cifrado en varias capas.

III. Análisis y enfoque NoSQL

Se puede identificar algunos problemas en las propuestas anteriores.

En el caso de los sistemas con cifrado en descanso (*at rest*, sección II-A), es necesario confiar en el proveedor puesto que es éste el que elige y custodia las claves de cifrado. Además, al ser sistemas de gestión de bases de datos puramente relacionales, no han sido diseñados para ser escalables bajo el modelo de computación en la nube; es sobre el cliente donde recae la responsabilidad de la escalabilidad, diseñando y adaptando el esquema de su base de datos de forma especial para poder escalar de forma limitada mediante particionamiento (*sharding*) u otras técnicas similares. No parece ser la mejor solución para el modelo *DaaS*.

Si bien la propuesta de Rodríguez (sección II-C) soluciona el problema de la confianza, puesto que es el cliente quien elige y custodia las claves de cifrado, presenta el mismo inconveniente frente a la escalabilidad horizontal del sistema. Por otra parte, el uso de criptografía homomórfica implica una gran carga computacional, si bien, en esta propuesta no se cuantifica suficientemente el impacto sobre el rendimiento del sistema no cifrado.

El sistema *CryptDB* (sección II-D) delega toda la gestión de seguridad al proxy traductor de consultas; por lo tanto, dicho proxy debe estar gestionado y alojado por el cliente para evitar tener que confiar en el proveedor *DaaS*. Bajo este modelo, el *back-end* relacional cifrado permite escalabilidad horizontal utilizando técnicas tradicionales como particionamiento (al igual que los sistemas con cifrado en descanso y la propuesta de Rodríguez) pero el proxy se convierte en un cuello de botella por el que pasan absolutamente todas las transacciones al *back-end* cifrado. Por otra parte, una gestión eficiente del proxy obligaría al cliente a tener y mantener su propio sistema de computación en cloud privado, que es, precisamente, lo que se quiere evitar al adoptar una estrategia *DaaS*.

Por todo lo anterior, consideramos que una posible solución a los sistemas *DaaS* seguros se encuentra en el uso de sistemas no exclusivamente relacionales (*NoSQL*, véase [1]), ya que están diseñados desde su origen para la escalabilidad horizontal en sistemas de computación en la nube.

Si bien existen funcionalidades equivalentes en ambos entornos, es necesario tener en cuenta que hay una serie de diferencias (véase [8]) entre ambos sistemas de bases de datos. En primer lugar, los datos no se almacenan en tablas sino en estructuras de documentos; no existen esquemas de tablas definidos en *NoSQL* ya que las estructuras pueden crecer de forma dinámica. Además, no existe un lenguaje estructurado de consultas estándar al estilo de *SQL* en los sistemas relacionales; podemos encontrar sistemas de bases

de datos *NoSQL* que tienen un lenguaje *UnQL* (Unstructured Query Language), pero la sintaxis difiere en cada una de ellas.

Los sistemas de bases de datos *NoSQL* están orientadas a cumplir el Teorema de Brewer o Teorema *CAP* (Consistency, Availability and Partition tolerance; véase [7]) y no el modelo *ACID* (Atomicity, Consistency, Isolation and Durability) propio de los sistemas relacionales transaccionales. No obstante, existen alternativas para solventar este inconveniente.

Al igual que en las propuestas de Rodríguez y *CryptDB*, el objetivo principal bajo el enfoque *NoSQL* consiste en que la base de datos pueda ser gestionada por un tercero manteniendo la confidencialidad de la información y permitiendo la realización de consultas sobre los datos como si estuvieran en claro. Para ello, resulta imprescindible el almacenamiento múltiple de cada campo con distintos tipos de cifrado en función de las operaciones a realizar sobre el mismo.

A continuación, detallamos algunas de las operaciones a considerar junto con sus posibles soluciones:

- *Alta, baja y modificación de documentos.* Es necesario que el cifrado se realice en el cliente, determinando los esquemas de cifrado oportunos en función del tipo de cada campo: cifrado homomórfico y cifrado que preserva el orden en el caso de campos numéricos y cifrado determinístico en caso de campos alfanuméricos.
- *Búsqueda de elementos.* Para el filtrado y localización de elementos se ha de considerar la naturaleza del campo sobre el que se está buscando para determinar que cifrado utilizar. Esta operación se realiza en el cliente.
- *Orden de elementos.* El uso de un cifrado que preserva el orden permite ordenar resultados y delimitar intervalos.
- *Suma de elementos.* Empleando un cifrador homomórfico para la suma, el servidor es capaz de realizar las operaciones de suma sin tener que descifrar los datos.
- *Relación entre documentos.* Consiste en la búsqueda de un elemento relacionado en dos colecciones de datos.

Por otra parte, es necesario encontrar una implementación adecuada para el cifrado de datos numéricos en coma flotante y de tipo fecha, de modo que se pueda operar con ellos en el servidor sin necesidad de descifrar los mismos.

IV. Conclusión

Se han descrito y analizado algunas propuestas significativas de sistemas de bases de datos seguros para la computación en la nube. También se ha introducido una posible solución a las deficiencias de los sistemas existentes mediante un enfoque no exclusivamente *SQL* (*NoSQL*).

Los sistemas *NoSQL*, al contrario que los sistemas de gestión de bases de datos relacionales tradicionales, han sido diseñados desde el origen para la computación en la nube; por lo tanto, posibilitan niveles de rendimiento y escalabilidad óptimos en dichas plataformas. No obstante, presentan ciertas dificultades como el hecho de almacenar la información en documentos sin estructura relacional o no ofrecer un lenguaje de consulta estándar como el *SQL*.

Por ello, este enfoque resulta una vía de trabajo futuro muy interesante que se está explorando en la actualidad.

Agradecimientos

Investigación parcialmente financiada por el MINECO mediante el proyecto TIN2011-25452.

Referencias

[1] L. Adam, J. Mattson, "Investigating storage solutions for large data: A comparison of well performing and scalable data storage solutions for real time extraction and batch insertion of data," master of science thesis, Department of Computer Science and Engineering, Chalmers University of Technology, Göteborg, Suecia, 2010.

[2] A. Alcalde, "Lo último en criptografía: Fully Homomorphic Encryption," 2013. Disponible en http://elbauldelprogramador.com/lo-ultimo-en-criptografia-fully-homomorphic-encryption

[3] A. Boldyreva, N. Chenette, Y. Lee, A. O'Neill, "Order-preserving symmetric encryption" en *Advances in Cryptology- EUROCRYPT 2009*, Praga, República Checa, LNCS vol.5479, 2009, pp. 224-241.

[4] C. Curino, E.-P.-C. Jones, R.-A. Popa, N. Malviya, E. Wu, S. Madden, H. Balakrishnan, N. Zeldovich, "Relational cloud: A database-as-a-service for the cloud," en *Proceedings of the 5th Biennial Conference on Innovative Data Systems Research*, Pacific Grove, CA, 2011, pp. 235–241.

[5] K. Van Gelder, "Elastic Data Warehousing in the Cloud," Report Faculty of Exact Sciences, Vrije Universiteit Amsterdam, Netherlands, 2011. Disponible en http://homepages.cwi.nl/~boncz/msc/2011-KeesvanGelder.pdf

[6] C. Gentry, "A fully homomorphic encryption scheme", tesis doctoral, Departament of Computer Science, Stanford University, 2009.

[7] S. Gilbert, N. Lynch, "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services," SIGACT News, vol. 33 Iss. 2, pp. 51–59, 2002.

[8] L. P. Issac, "SQL vs NoSQL Database Differences Explained with few Example DB," The Geek Stuff, 2014. Disponible en http://www.thegeekstuff.com/2014/01/sql-vs-nosql-db/

[9] Microsoft docs., "Cifrado de datos transparente (TDE),". Disponible en http://technet.microsoft.com/es-es/library/bb934049.aspx

[10] Oracle docs., "Transparent Data Encryption," Oracle Database Advanced Security Administrator's Guide. Disponible en http://docs.oracle.com/cd/B19306_01/network.102/b14268/asotrans.htm

[11] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," en *Advances in Cryptology- EUROCRYPT 2009*, Praga, República Checa, LNCS vol.1592, 2009, pp. 223-238.

[12] R. A. Popa, N. Zeldovich, H. Balakrishnan, "CryptDB: A Practical Encrypted Relational DBMS," Report MIT-CSAIL-TR-2011-005, Computer Science and Artificial Intelligence Laboratory, Cambridge, 2011.

[13] R. A. Popa, C. Redfield, N. Zeldovich, H. Balakrishnan, "Cryptdb: protecting confidentiality with encrypted query processing," en *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, Cascais, Portugal, 2011.

[14] R. A. Popa, N. Zeldovich, "Cryptographic treatment of CryptDB's Adjustable Join," Report MIT-CSAIL-TR-2012-006, Computer Science and Artificial Intelligence Laboratory, Cambridge, 2012.

[15] R. A. Popa, F. H. Li, N. Zeldovich, "An ideal-security protocol for order-preserving encoding," en *Proceedings of 2013 IEEE Symposium on Security and Privacy*, 2013, pp. 463–477.

[16] L.M.X. Rodríguez, "Esquema de cifrado para la ejecución de consultas en bases de datos cifradas," tesis doctoral, Departamento de Computación del Centro de Investigación y Estudios Avanzados del Instituto Politécnico Nacional, México, D.F., Diciembre, 2009.

[17] S. Tu, M. F. Kaashoek, S. Madden, N. Zeldovich, "Processing analytical queries over encrypted data," en *Proceedings of the 39th international conference on Very Large Data Bases*, Riva del Garda, Italia, 2013, pp. 289–300.