PRISMA/DB Technical Overview

Prisma/DB is a database encryption solution that provides per-attribute encryption granularity and enables operations over encrypted columns without *ever* sharing private keys with the database server. Prisma/DB is a result of 6 years of meticulous research done within Cyber Security Lab of Nanyang Technological University, Singapore. Prisma/DB is able to provide this functionality by putting together many well-established cryptosystems and secure data processing techniques and making them work together in a seamless way. Leveraging the deep integration with the query language and the database protocol, Prisma/DB is fully transparent for the application and could be installed in a plug-and-play manner.

There is a multitude of database security solutions on the market. An absolute majority of database encryption products available today provide a so-called Transparent Data Encryption (TDE): the database server encrypts data before it is written to disk, and decrypts it whenever it is read again. While efficient and simple, TDE has two major drawbacks: 1) it only protects data-at-rest, leaving out data-in-use and data-in-motion; 2) it requires that the private key is stored within the database server—a critical issue for over 60% of polled institutions, according to 2015 Cloud Security Alliance report. Another solution that is available on the market today is a feature of MS SQL Server 2016+ called “Always Encrypted”. However, while “Always Encrypted” is built on similar principles as Prisma/DB, it provides much narrower capabilities while requiring much higher integration costs.

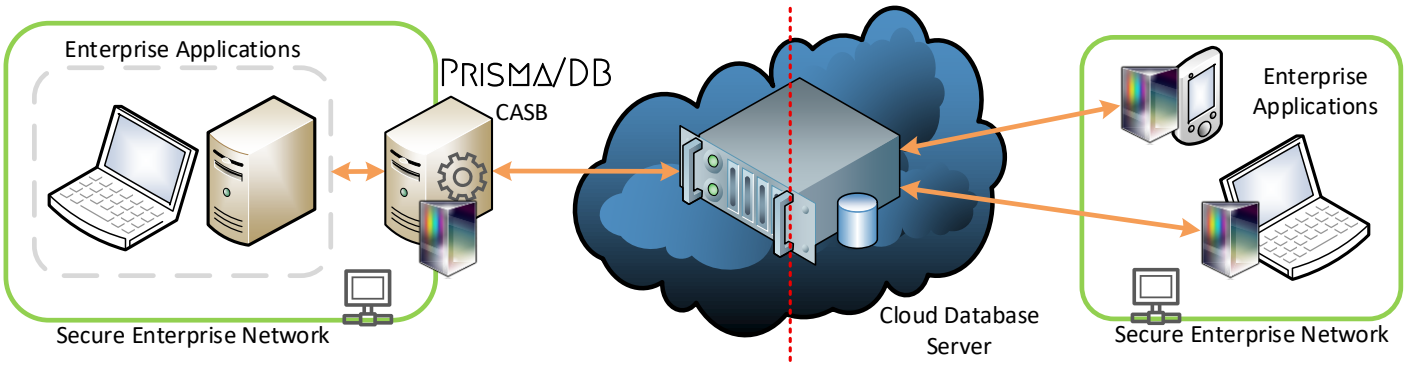

Figure 1. Two modes of operation. As a CASB (or a proxy, left) or as a software library (right).
Private keys never leave secure perimeter.

Prisma/DB comes in two “flavors”: as a stand-alone transparent cloud access security broker (CASB, or simply a proxy), and as a software library. The proxy holds the private and public keys, intercepts queries from applications to the database, encrypts them and sends to the database server. When the database server sends an encrypted result of the query, the proxy intercepts it, decrypts, and hands back to the application. By fully implementing the database network protocol, the proxy does not require any changes to the application code. The software library “flavor” can provide a tighter coupling and alleviate unnecessary overhead but requires minor changes to the applications to use the library instead of the regular database client driver libraries.

Currently, Prisma/DB only works with MS SQL Server and MySQL/MariaDB; the library “flavor” can only be used in .NET languages (C#, F#, C++.NET, VB.NET, IronPython, and some others). There is currently an active development towards supporting languages with native interfaces, including C/C++, Java, Python, etc. Future plans involve expanding to also support Oracle database and PostgreSQL. Prisma/DB currently leverages AES, ElGamal and Paillier cryptosystems and HMAC-based hashes, but could be easily extended to use any other cryptosystem (e.g., Gentry’s homomorphic scheme) due to its modular design.

Developers’ section of the website (http://prismadb.com/) contains links to several demos of the product, and detailed instructions on how to run them.