Examining Leakage from Access Counts in ORAM Constructions

Nikolaos P. Karvelas, Amos Treiber, and Stefan Katzenbeisser

Technische Universität Darmstadt, Germany

29th Crypto Day, 6/7 September 2018

Oblivious RAM (ORAM), introduced by Goldreich & Ostrovsky (1996), is a cryptographic primitive that provides access pattern hiding for outsourced databases, meaning that the server cannot distinguish between any two possible database accesses by the client based on the observed server-side memory instructions. Compared to conventional outsourced data privacy using just encryption, the relatively strong privacy guarantee here is that the server cannot acquire any knowledge about *which* database item was accessed. However, there does not seem to be a construction that is safe against the potential leakage due to knowledge about the mere *amount* of accesses performed by a client. We consider this a violation of privacy, as user data is often stored in a domain-specific manner. For example, using an ORAM one can hide which genes are of interest in the domain of *in silico* medical genetic testing. But, since certain symptoms correspond to different amounts of genes that have to be queried, the amount of accesses on such a database might leak information about a patient's medical concerns.

To our knowledge, this issue has not been formally addressed in the literature so far. In this work, we intend to close this gap by investigating the leakage vector in a formal setting independent of the use case and by examining all popular constructions in this regard. Contrary to the classical ORAM adversarial model, we consider an adversary that does not have access to the network communication but instead can regularly obtain copies ('snapshots') of the ORAM database at any point in time. Although our adversary seems rather weak, we argue that it still is realistic for the scenario, given that outsourced data can often be obtained via theft, data-breaches, or even backdoors. We propose the game-based definition of *indistinguishability of number of accesses* (IND-NOA). The adversary receives an ORAM snapshot taken after one of the two amounts of accesses specified in the adversary's challenge has been performed by the client. If any adversary correctly guesses which amount was chosen with negligible advantage, then we call the ORAM IND-NOA secure or 'access count private'.

Using the IND-NOA definition, we examine the privacy of all popular constructions. We formally prove that TrivialORAM, where the client just scans and re-encrypts the entire database, is access count private. Moreover, we show that the more efficient non-trivial constructions employed today (SquarerootORAM

crypto day matters 29 (2018)
Beiträge vom 29. Krypto-Tag, 06.-07.09.2018, Bosch, Renningen.
doi:10.18420/cdm-2018-29-24

and HierarchicalORAM by Goldreich & Ostrovsky (1996), and PathORAM by Stefanov, van Dijk, Shi, Fletcher, Ren, Yu & Devadas (2013)) are not access count private. We use the strategies of the presented adversaries in a general manner to argue that, given the way all non-trivial ORAMs are constructed so far, it *seems* impossible that any non-trivial construction can be access count private.

Based on these results, it seems unlikely that one can fully avoid access count leakage in practice. Hence, we also move outside of formal security by deriving privacy metrics concerning access counts for all popular constructions. The results represent a quantification of which architectures offer the best privacy in practice: the metrics for SquarerootORAM and HierarchicalORAM are relatively close to the one for TrivialORAM, while the metric for PathORAM indicates significantly worse access count privacy.

References

Oded Goldreich & Rafail Ostrovsky (1996). Software Protection and Simulation on Oblivious RAMs. *Journal of the ACM* **43**(3).

Emil Stefanov, Marten van Dijk, Elaine Shi, Christopher W. Fletcher, Ling Ren, Xiangyao Yu & Srinivas Devadas (2013). Path ORAM: an extremely simple oblivious RAM protocol. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM.