МОДЕЛИРОВАНИЕ СИСТЕМЫ ЗАЩИТЫ ИНТЕРФЕЙСА ПЕРЕДАЧИ ДАННЫХ ПРИ РАБОТЕ С СЕТЕВЫМИ СЕРВИСАМИ НА ОСНОВЕ КОНЦЕПЦИИ «OBLIVIOUS RAM»

Макаров А.Е.[1], Варламов А.А.[2]

[1]Макаров Анатолий Евгеньевич - старший архитектор программного обеспечения,
DigitalIQ, Феникс, штат Аризона;
[2]Варламов Александр Александрович - старший архитектор решений,
LI9 INC,Роли, Северная Каролина,
Соединенные Штаты Америки

Аннотация: *рассмотрены особенности организации интерфейса безопасной передачи данных при работе с аппаратно-программной платформой облачного сервиса. Указаны преимущества использования интерфейсов класса «Oblivious RAM» с целью защите «чувствительных данных» пользователя в том случае, когда владелец сервиса придерживается политики «Honest but Curious». Предложена модель построения древовидной структуры канала передачи данных, которая базируется на оптимизации эффективности работы алгоритмов защиты данных через варьирование количества дочерних узлов и промежуточных уровней. Целевыми функциями определения эффективности интерфейса «Oblivious RAM» выступали показатели нагрузки на пропускную способность информационного канала и ресурс информационного хранилища сервера, а также показатель надежности защиты данных. Проведена оценка эффективности разработанного алгоритма по сравнению с алгоритмом «Partition ORAM».*

Ключевые слова: *облачный сервис, «чувствительные данные», интерфейс «Oblivious RAM», информационное хранилище, пропускная способность, математическая модель, целевая функция.*

Введение

Последнее десятилетие охарактеризовалось экспоненциальным ростом количества и расширением функционала центров обработки данных (ЦОД) на базе облачных сервисов. На сегодняшний день в связи с организацией глобальных информационных сетей, каналы передачи данных которых характеризуются высокими показателями пропускающей способности, в области услуг облачных сервисов актуализировалась модель «Platform as a Service» (PaaS). В рамках PaaS клиент арендует аппаратно-программную платформу ЦОД, которая характеризуется оговоренными объемом информационного хранилища и вычислительным ресурсом, а также, при необходимости, соответствующим программным обеспечением: операционной системой (ОС) и специализированными программными приложениями. Очевидно, что указанный подход потенциально небезопасен для клиента, который передает в инфраструктуру ЦОД т.н. «чувствительные данные», утечка которых может представлять репутационные потери, упущенную коммерческую выгоду либо стратегическую опасность для государственных структур и частных военных компаний, являющихся клиентами облачного сервиса. Разумеется, на уровне самого ЦОД регулярно внедряются и совершенствуются методы защиты от утечки данных (Data Leakage Protection, DLP), однако одновременно, это приводит к тому, что владелец сервиса на базе системы мониторинга собирает значительный массив данных о деятельности клиента [1-5]. Это может проводиться как в рамках протокола класса «Honest but Curious» (т.е., собирается максимум информации о клиенте из числа доступной в рамках договора) так и через прямое нарушение политики соблюдения конфиденциальности. Таким образом, в случае противоправных действий со стороны персонала ЦОД либо внешней атаке на сервис «чувствительные данные» могут оказаться у злоумышленников.

В связи с этим значительную актуальность приобретает методика защиты «чувствительных данных» на физическом уровне использования аппаратной платформы серверного комплекса ЦОД путем разбиения передаваемого набора данных на блоки одного размера, а также создания фиктивных блоков данных, которые также передаются через информационные каналы и хранятся в информационном хранилище ЦОД. Т.к. исторически указанный подход подразумевает организацию защищенного интерфейса при работе центрального процессора (Central Processing Unit, CPU) с ячейками операционной памяти (Random-Access Memory, RAM), он был назван «Oblivious RAM» [6-8]. Но современные модели «Oblivious RAM» в большей степени базируются на сокрытии процессов загрузки пользователем данных со своей рабочей станции на сервер ЦОД (аналог процесса записи) и загрузки пользователем данных с сервера ЦОД на свою рабочую станцию (аналог процесса считывания).

При организации «Oblivious RAM» создание фиктивных блоков данных и усложнение пути передачи данных с целью сокрытия от владельца сервиса последовательности операций загрузки данных на сервис и с сервиса значительно увеличивается нагрузка на информационный канал и информационное хранилище сервиса. Анализ ***современных исследований и публикаций в профильных изданиях*** посвященных данной проблеме представляет набор подходов по оптимизации схем внедрения и настройки интерфейса «Oblivious RAM» в зависимости от постановки задачи. Наиболее актуальные из них можно разделить на следующие группы:

- схемы построения интерфейса класса «Burst ORAM» [9, 10];
- схемы построения интерфейса класса «Partition ORAM» [11, 12];
- схемы построения интерфейса класса «Ring ORAM» [13, 14];
- схемы построения интерфейса класса «Path ORAM» [15-17];
- схемы построения интерфейса класса «Binary Tree ORAM» [18-20] .

Представленные результаты указывают на возможность уменьшить нагрузку на информационный канал при стабильном уровне защиты данных, в то время как использование ресурса информационного хранилища остается чрезмерной. В случае хранения большого объема данных это может привести к существенному удорожанию использования услуг PaaS, что выделяется как ***нерешенная часть*** вопроса в рамках общего исследования.

Целью работы, таким образом, стала разработка комплексной методики построения интерфейса «Oblivious RAM», которая позволяет уменьшить нагрузку как на информационный канал, так и на ресурс информационного хранилища при стабильном уровне защиты данных. Методика базируется на схеме «Binary Tree ORAM», но при этом предлагается использовать небинарное дерево канала передачи данных.

1. Постановка задачи оптимизации интерфейса «Oblivious RAM» с целью уменьшения нагрузки на платформу облачного сервиса

Постановка задачи оптимальной организации интерфейса «Oblivious RAM» с целью уменьшения нагрузки на пропускающую способность информационного канала и ресурс информационного хранилища инфраструктуры облачного сервиса базируется на построении базовой модели интерфейса и выделении целевых функций эффективности его функционирования.

В основе базовой математической модели лежит представление передаваемого блока данных A_i в направлении «пользователь → сервис» либо «сервис → пользователь» как следующего множества элементов:

$$A_i: \{T_i, P_i\}, где F_i = \{t_i^n\} для n \in [1; N_i], а F_i = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, (1)$$

где: $i \in [1; I]$ — уникальный номер (идентификатор) блока данных, T_i — блок данных (N_i — количество символов блока i, t_i^n — n-ый символ блока i), F_i — идентификатор операции ($F_i = 0$ соответствует операции передачи данных в направлении «пользователь → сервис», а $F_i = 1$ соответствует операции передачи данных в направлении «сервис → пользователь»).

Рассмотрим критерий защищенности интерфейса передачи данных в соответствии с протоколом передачи «Oblivious RAM». Если у нас есть два блока данных A_i и A_j где $i, j \in [1; I]$ реальных (служащих для обмена информацией) либо фиктивных (служащих для сокрытия процесса передачи данных). Пусть $F_S(\quad)$ — функция работы с данными, результаты выполнения которой доступны на стороне облачного сервиса. В таком случае интерфейс передачи данных является защищенным, если при $N_i = N_j$ для $\forall i, j \in [1; I]$ — $F_S(A_i) = F_S(A_j)$.

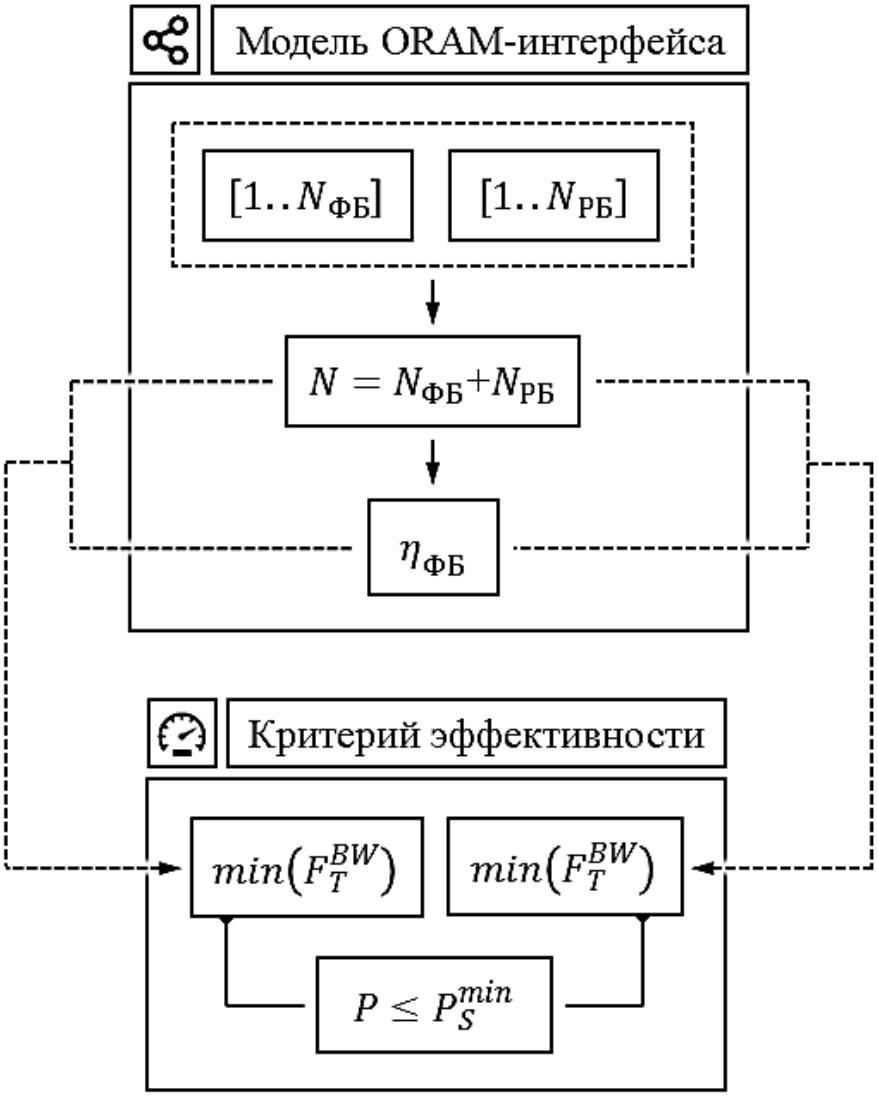

Рис. 1. Схема оптимизации и оценки эффективности интерфейса передачи на данных «Oblivious RAM».

Уровень защиты «чувствительных данных» при этом определяется через вероятность P_S воспроизведения последовательности из операций передачи полезной информации путем перебора возможных вариантов и, таким образом, зависит от процента фиктивных блоков (ФБ) по сравнению с реальными блоками (РБ), а также организации пути передачи данных. В то же время увеличение процента ФБ ($\eta_{ФБ}$) в общем случае приводит к увеличению нагрузки на информационный канал и ресурс информационного хранилища. Задача оптимизации, соответственно, может быть решена через нахождение глобальных минимумов целевых функций нагрузки на канал передачи данных (F_T^{BW}) и ресурс информационного хранилища (F_T^{IS}) при необходимом уровне защиты данных P_S^{min}:

$$\begin{cases} min(F_T^{BW}) \\ min(F_T^{IS}) \end{cases} при P \leq P_S^{min}. (2)$$

Аргументами целевых функций являются количество ФБ ($N_{ФБ}$) и РБ ($N_{РБ}$), а также организация пути передачи данных, что будет рассмотрено ниже.

2. Построение интерфейса «Oblivious RAM» на базе архитектуры небинарного дерева

Построение интерфейса «Oblivious RAM» на базе деревоподобной структуры пути передачи данных подразумевает, что каждый узел, начиная с узла рабочей станции пользователя и исключая узлы информационного хранилища, имеет набор дочерних узлов $k \in [1; K]$. Для бинарного дерева значение $K = 2$, а для небинарного — $K > 2$, но также является фиксированным. Для построения математической модели интерфейса «Oblivious RAM» на базе архитектуры небинарного дерева необходимо ввести следующие обозначения:

- уровни пути передачи данных $l \in [1; L]$;
- корневой узел, на котором находится рабочая станция клиента облачного сервиса ($l = 1, i = 1$), на данном уровне не содержатся ФБ;

- граничные узлы, на которых располагается информационное хранилище ($l = L, i \in [(K + K^2 \ldots + K^{L-1}); (K + K^2 \ldots + K^L)]$);
- промежуточные узлы, которые располагаются между корневым узлом и граничными узлами ($l \in [2; L-1], i \in [2; (K + K^2 \ldots + K^{L-1} - 1)]$).

С целью оптимизации интерфейса «Oblivious RAM» путем расчета оптимальных значений K, L и $\eta_{ФБ}$ необходимо ввести следующие показатели:

- процент РБ в промежуточных узлах $\eta^{П}_{РБ}$;
- процент ФБ в промежуточных узлах $\eta^{П}_{ФБ}$;
- процент РБ в граничных узлах $\eta^{Г}_{РБ}$;
- процент ФБ в граничных узлах $\eta^{Г}_{ФБ}$;
- информационная емкость узла S;
- коэффициент вычисления граничных значений доступной информационной емкости узла $\kappa = (K-1)/2$;
- объем данных, который хранится на граничном узле $S^{Г}_{\Sigma}$ и состоит из РБ ($S^{Г}_{РБ}$) и ФБ ($S^{Г}_{ФБ}$), где $S^{Г}_{\Sigma} = S^{Г}_{РБ} + S^{Г}_{ФБ}$;
- объем данных, который хранится на промежуточном узле $S^{П}_{\Sigma}$ и состоит из РБ ($S^{П}_{РБ}$) и ФБ ($S^{П}_{ФБ}$), где $S^{П}_{\Sigma} = S^{П}_{РБ} + S^{П}_{ФБ}$.

Для вычисления оптимальных значений граничных и промежуточных узлов ($S^{Г}_{\Sigma}$ и $S^{П}_{\Sigma}$, соответственно), а также количества уровней, предлагается использовать следующие уравнения:

$$L = \lfloor log_k \left(\frac{I}{\kappa S}\right) \rfloor + 2, (3)$$

$$S^{Г}_{\Sigma} = \frac{(1 + \eta^{П}_{ФБ}) \cdot S^{Г}_{РБ}}{\lfloor \frac{S^{Г}_{РБ}}{\kappa S} \rfloor}, (4)$$

$$S^{П}_{\Sigma} = \kappa S \cdot (1 + S^{П}_{РБ}) + S \cdot (1 + S^{П}_{ФБ}). (5)$$

Результаты использования предложенного алгоритма по сравнению с алгоритмом «Partition ORAM» (были использованы открытые статистические данные исследовательской группы Кюмао Ма [21]) представлены на рис. 2 и 3.

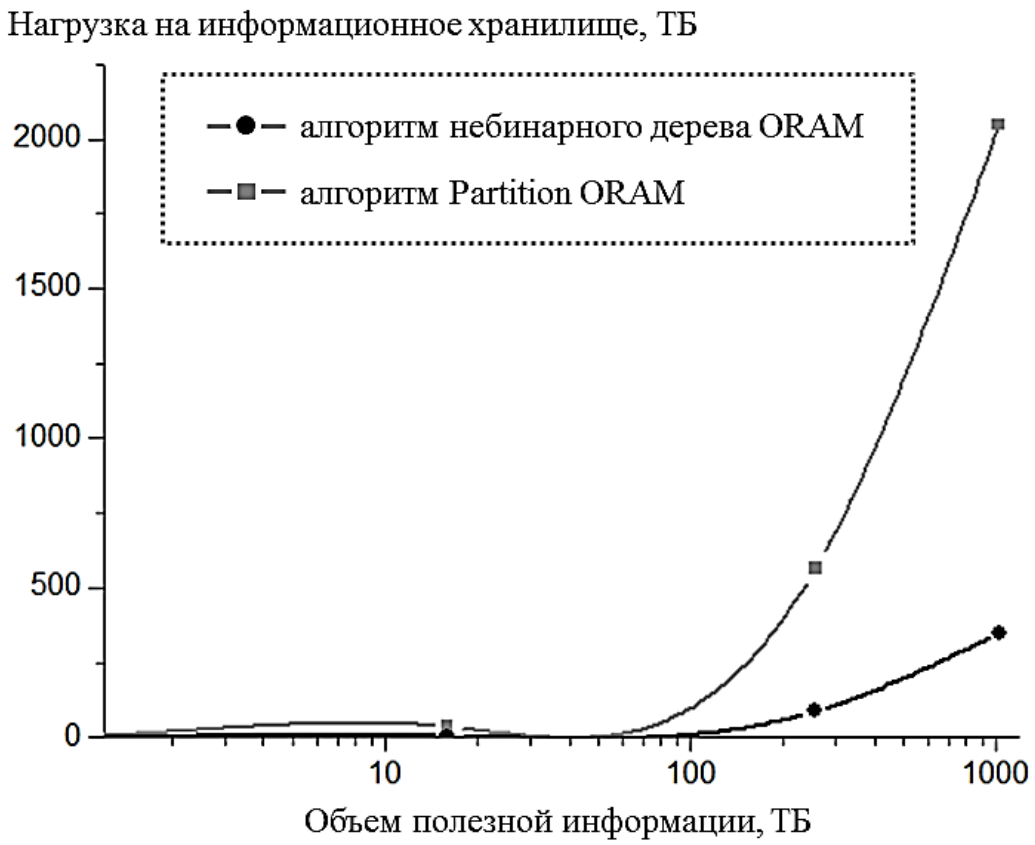

Рис. 2. Расчет нагрузки на информационное хранилище облачного сервиса для алгоритмов на основе небинарного дерева и «Partition ORAM»

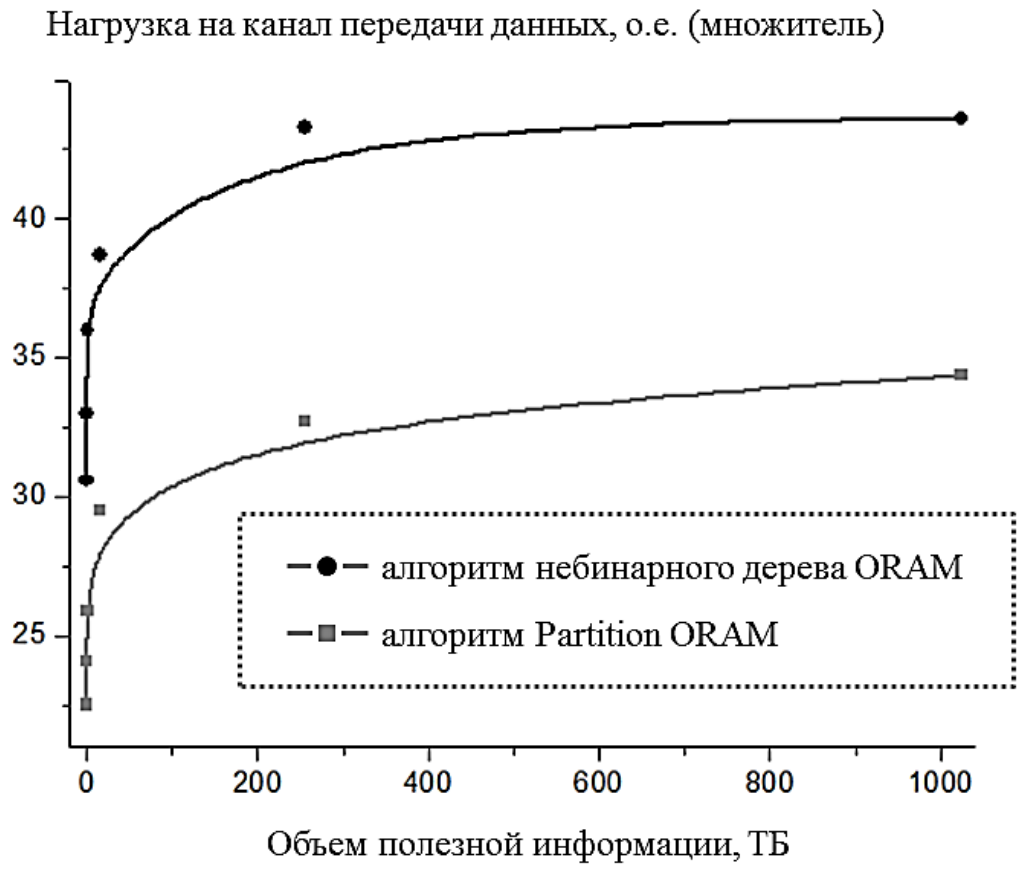

Рис. 3. Расчет нагрузки на канал передачи данных облачного сервиса для алгоритмов на основе небинарного дерева и «Partition ORAM»

Результаты статистического анализа показывают, что алгоритмы на основе небинарного дерева дают существенное преимущество при хранении больших объемов данных, но при этом нагрузка на канал передачи данных увеличивается в 1,3 - 1,4 раза.

Выводы

В результате проведенной работы были рассмотрены особенности организации безопасной передачи «чувствительных данных» при использовании аппаратной платформы облачного сервиса. Указаны преимущества применения в данной области интерфейсов класса «Oblivious RAM», а также на основе математической модели показан рост нагрузки на канал передачи данных и информационное хранилище. Предложена схема структуры канала передачи данных на основе небинарного дерева, целевыми функциями которой выступали показатели нагрузки и надежности защиты данных. Проведена оценка эффективности разработанного алгоритма по сравнению с алгоритмом «Partition ORAM», которая показала преимущества алгоритма на основе небинарного дерева в случае необходимости хранения больших объемов данных.

Список литературы

1. *Yuan D.* (2013). Computation and storage in the cloud understanding the trade-offs. Elsevier.
2. *Foster I. & Gannon D.B.* (2017). Cloud computing for science and engineering. The MIT Press.
3. *Chaki R.* (2015). Applied computation and security systems. Springer.
4. *Wang G., Chen B., Li W., Pietro R.D., Yan X. & Han H.* (2021). Security, privacy, and anonymity in computation, communication, and storage: 13th International Conference, SpaCCS 2020, Nanjing, China, December 18-20, 2020: proceedings. Springer.
5. *Smith H.* (2011). Data center: cost-effective strategies, implementation, and management. CRC Press.
6. *Cho S., Hwang S., Shin W., Kim N. & In H.P.* (2021). Design of Military Service Framework for Enabling Migration to Military SaaS Cloud Environment. Electronics, 10 (5). 572.
7. *Zhang J., Ma Q., Zhang W. & Qiao D.* (2017). TSKT-ORAM: A Two-Server k-ary Tree Oblivious RAM without Homomorphic Encryption. Future Internet, 9 (4). 57.
8. *Hoang T., Yavuz A.A. & Guajardo J.* (2020). A Multi-server ORAM Framework with Constant Client Bandwidth Blowup. ACM Transactions on Privacy and Security, 23 (1). 1–35.
9. *Dautrich J. & Ravishankar C.* (2015). Combining ORAM with PIR to Minimize Bandwidth Costs. Proceedings of the 5th ACM Conference on Data and Application Security and Privacy. https://doi.org/10.1145/2699026. 2699117.
10. *Dautrich J. & Ravishankar C.* (2015). Tunably-Oblivious Memory. Proceedings of the 5th ACM Conference on Data and Application Security and Privacy. https://doi.org/10.1145/2699026.2699097.

11. *Zhang J., Zhang W. & Qiao D.* (2015). GP-ORAM: A Generalized Partition ORAM. Network and System Security, 268–282. https://doi.org/10.1007/978-3-319-25645-0_18.
12. *Zhang J., Zhang W., Qiao D.* (2014) S-ORAM: a segmentation-based oblivious RAM. In: Proc. ASIACCS.
13. *Ren L., Fletchery C.W., Kwony A., Stefanov E., Shi E., van Dijkz M. and Devadasy S.* (2014) Ring ORAM: Closing the Gap Between Small and Large Client Storage Oblivious RAM. In Proc. IACR Cryptology ePrint Archive 2014: 997.
14. *Boneh D., Mazieres M. and Popa R.A.* Remote oblivious storage: Making oblivious RAM practical. Manuscript, http://dspace.mit.edu/bitstream/ handle/1721.1/62006/MIT-CSAIL-TR-2011-018.pdf, 2011.
15. *Stefanov E., van Dijk M., Shi E., Fletcher C., Ren L., Yu X. & Devadas S.* (2013). Path ORAM. Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security - CCS '13. https://doi.org/10.1145/ 2508859.2516660.
16. *Ren L., Yu X., Fletcher C.W., van Dijk M., Devadas S.* (2006) Design space exploration and optimization of path oblivious RAM in secure processors. In: Proc. ISCA.
17. *Wang R., Zhang Y. & Yang J.* (2018). D-ORAM: Path-ORAM Delegation for Low Execution Interference on Cloud Servers with Untrusted Memory. 2018 IEEE International Symposium on High Performance Computer Architecture (HPCA). https://doi.org/10.1109/hpca.2018.00043.
18. *Zhang J., Ma Q., Zhang W. & Qiao D.* (2016). TSKT-ORAM: A two-server k-ary tree ORAM for access pattern protection in cloud storage. MILCOM 2016 - 2016 IEEE Military Communications Conference. https://doi.org/ 10.1109/ milcom.2016.7795381.
19. *Li S., Zhao M., Jiang H., Xu Q. & Wei X.* (2016). Efficient ORAM Based on Binary Tree without Data Overflow and Evictions. Communications in Computer and Information Science, 596–607. https://doi.org/10.1007/978-981-10-0356-1_62.
20. *Li S., Zhao M., Jiang H., Xu Q.,& Wei,X.* (2016). Efficient ORAM Based on Binary Tree without Data Overflow and Evictions. Communications in Computer and Information Science, 596–607. https://doi.org/10.1007/978-981-10-0356-1_62.
21. *Ma Q. & Zhang W.* (2018). Towards Practical Protection of Data Access Pattern to Cloud Storage. MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM). https://doi.org/10.1109/milcom.2018. 8599798.