Mix-ORAM

Raphael R. Toledo [*]

University College London

May 19, 2016

Abstract

We present in this paper an ORAM system relying on mixnet technologies for the eviction process.

1 Introduction

Most of existing ORAM solutions have so far required the whole database to be periodically rerandomized by the user. In 2013, Stefanov and Shi [] proposed a multi-cloud oblivious storage where two or more semi-trusted clouds are used to serve user requests and hide the access pattern. However, the practicality of owning and managing more than a cloud was not discussed in the paper which presented. In this work, we suggest to leave the eviction and shuffling to an untrusted third party and attest its privacy thanks to an information theoric proof. The advantages of such use are the delegation of the shuffling, the possibility to postpone the latter to quieter times and the non dependance to a centralized party.

1.1 Related Work

ORAM.
Mixnets.
Permutation. Random transpositions shuffle (RTS) have been thoroughly studied since 1981 when Diaconis and Ashahshahani released their paper on random permutation generations cite. RTS consists in randomly selecting two cards from a deck of cards and exchange them if they are different. The time of convergence towards the uniform distribution, the mixing time, was bounded at $(2 + o(1))\, n \log(n)$ thanks to strong a stationary time argument due to Broder[]. Diaconis however observed a cutoff phenomenon in his first paper in 1981, an abrupt convergence to the equilibrium distribution, at $\frac{1}{2} n \log(n)$ of width $O(n)$.

[*]E-mail: r.toledo@cs.ucl.ac.uk

1.2 Our contributions

1.3 Content

2 Preliminaries

2.1 Model

We consider a system composed of a remote server running the ORAM protocol with storage of size $O(N)$ where N corresponds to the number of b bits cipher blocks the user owns and a mixnet of m mixes. The client includes a storage manager and a stash of s-block capacity.

2.2 Threat model

We consider an adversary $\mathcal{A}$ has corrupted m_a mixes and potentially the ORAM database in order to discover the relationship between the records and the indexes. The corrupted servers are said honest but curious in that they perform correctly the assigned protocols and algorithms but passively record and share all information they observe to achieve this objective. We also assume that the adversary observes all the user's incoming and outgoing communication. However, we assume that all messages, database request or mix packet, are encrypted and that only message timing, volume and size from honest servers are visible to the adversary. Attacks and countermeasures, together with the implied costs, will only be examined in the Discussion section (c.f. Section.[]).

2.3 Costs

This work studies PIR scalability, and we focus on analyzing costs on the server side, which is the performance bottleneck of current techniques. We denote the communication costs by C_d for the number of *bits* sent by the ORAM, C_c by the client to $\mathcal{DS}$, and by C_m sent by the mixes. We will call C_p the computation cost derived from the mixes' permutation and C_a from the database accesses.

3 Mix-ORAM

Reencryption network vs Decryption network. Reencryption ORAM will be used as decryption requires the user to update all records with all the keys... (except two layer encryption?). Symmetrical Reencryption network ?

3.1 Eviction process: the Mixnet

To evict the stash, the client updates the former in the database and sends a signal to the mixnet or to the database if he wants the eviction to be postponed to quieter times.

Two different methods can be applied for the eviction process depending on the mixnet architecture, in both cases eviction rely on repeatedly fetching k records, refreshing and permuting them before updating them back in the ORAM. If the mixnet is a cascade network, the first mix outputs the resulting array to next mix while repeating the process for distinct records, otherwise, in the mesh case, the mixes fetch in parallel the records. The cascade case is more advantageous in time as the outputs and inputs of the mixes coincide, hence the communication cost is roughly divided by two. There is futhermore no synchronization needed between the mixes to process different records at the same time. However what is gain in duration is loose in resilience as any mix failure in the cascade mixnet would compromise the whole system. Moreover an attack specific to cascade mixnet also needs to be taken into account: if the first mix and the database are both corrupted, the permutation could be done only on a subset of records so that the adversary knows the positions of a non trivial number of records, as they were not permuted, without the honest mixes and the client realizing it. To counter it, the first mix query should be sent along the records to all the mixes together with an identification proof.

Use of distributive encryption ?

3.1.1 k-Random Transposition Shuffle

We argue that integrating k RTS, that we call k-permutation for short, in the ORAM access method could help the randomization of the database. The eviction process could also be postpone to minimize ORAM response time in a similar way as Burst ORAM [].

Security Theorem 1. *A k-permutation shuffle of a n card game reaches the uniform distribution in τ rounds, such that*

$$\begin{aligned} E(\tau) &< \frac{2}{k} \cdot \frac{n^2}{n+1} \cdot (\log(n) + \mathcal{O}(1)) \\ \sigma^2(\tau) &< \frac{4}{3} \cdot \pi^2 \cdot \left(\frac{n}{k}\right)^2 \\ \tau(\epsilon) &\geq \frac{n-1}{2k} \ln n \cdot \frac{1-\epsilon}{6} \end{aligned}$$

The computation cost of the eviction would thus be of the order of $m \cdot n \cdot \log(n)$ while the communication cost will be either $4 \cdot m \cdot n \log(n)$ or $2 \cdot (m+1) \cdot n \log(n)$ depending on the mixnet architecture.

4 Lowering Eviction Costs.

In order to lower the eviction costs sustained by the mixnet, two optimizations can be enforced

4.1 Client Optimization

4.1.1 User participation

The client, to lower the eviction costs, could instead of fetching the only the requested record query $k-1$ other records. Doing so, we can advance further towards the uniform distribution before the eviction process, measure the distance between the two distributions and reduce the database shuffling cost by having the mixnet performing cheaper and weaker permutations. What's more, requesting more records can help in hidding the access pattern to the adversary has discussed in [**?**].

For instance if $n = 10^4,\ k = \log(n) = 4$ and the stash has a size of $\sqrt(n) = 100$, without considering fake accesses, we need 25 accesses before evicting the stash.

4.1.2 Fake access

When a user wants to access a record which already is in the stash, a random database access is performed to hide potential frequency attacks: an adversary having observed the record access frequency beforehand and knowing the average access per day would be able to guess which records are in the stash by not observing accesses which should happen. These fake accesses can be considered as free permutations as no fetched records need to remain in the stash, thus we can arbitrarily set $k_{in} = k_{out}$.

Security Theorem 2. *In the case of a uniform distribution access we have an average number 0 of fake access for $s << n$.*

However it is well known that the frequency access of records is not uniform.

Security Theorem 3. *Let A be the access distribution. Let's call v_0 and v_1 the volume above the line $y = 0$ and under it respectively. The average number of fake access is $\frac{v0}{v1}$.*

Proof. The volume under such line represents the number of real accesses a user does while the volume above it represents accesses in the stash which leads to fake accesses. When the number of accesses is great, the access frequency approaches the average number of accesses. □

4.2 Mixnet Optimization

Instead of requiring the mixes to permute all of the k fetched records, the user could set as a parameter of the system the probability of permuting an element $p = \frac{\alpha}{k}$, with α the average number of records to permute. In that case, the probability a record's index is known to the adversary can be reduced to the probability a record has not been permuted.

Security Theorem 4. *The probability a record was not permuted is*

$$\Pr = \left(1 - \frac{\alpha}{k}\right)^{m-m_a} \quad \textit{during one round and,}$$
$$\Pr = \left(1 - \frac{\alpha}{k}\right)^{(m-m_a)\log(n)} \quad \textit{for the eviction.}$$

5 Evaluation

6 Discussion

Using proof of shuffle and proof of work to counter injection attacks. Using a reputation system alongside to dismiss mixes more quickly.

7 Conclusion

ORAM	Access Overhead	Message size	Client Memory	Server Storage
Others				
Mix	$\mathcal{O}(1)$	$\mathcal{O}(log(n))$	s	$\mathcal{O}(n)$
Mix opt	$k \cdot \log(n)$	$\mathcal{O}(k \cdot \log(n))$	s	$\mathcal{O}(n)$

Table 1: System comparison where the online access overhead is the number of accesses to retrieve the requested item.

A Proofs of Theorems

A.1 Proof of Permutation ORAM

A.1.1 Proof of k permutation

Proof. To first prove the upper bound and variance, we use Diaconis[] results which states that τ defined in the following game is a strong stationary time. In a random transposition shuffle, the cards chosen by the right and left hands at time t are respectively called R_t and L_t. Assuming that when $t = 0$, no card is marked, we mark R_t if R_t was unmarked before and either L_t is marked or $L_t = R_t$. The variable τ represents the time when every card has been marked, we call it the stopping time.

Let be τ_t the number of transpositions after the t^{th} card is marked, up to and including when the $(t+1)^{th}$ is marked.

$$\tau = \sum_{i=0}^{n-1} \tau_i$$

.

The τ_t are independant geometric variables with probability of success p_t as implied by the game rules. The probability of success corresponds to the probability of marking at least one card, one to t cards exactly. To do so, the right cards must be chosen from the unmarked set, comprising $n - t$ cards at time t, and the left cards from the union of the marked set and the right cards.

$$\begin{aligned}
p_t &= \sum_{i=1}^{min(k,n-t)} \binom{k}{j} \cdot \frac{\binom{t+j}{j} \cdot \binom{n-t}{j}}{\binom{n}{j}^2} \\
&= k\frac{(t+1)\cdot(n-t)}{n^2} + \sum_{i=2}^{min(k,n-t)} \binom{k}{j} \cdot \frac{\binom{t+j}{j} \cdot \binom{n-t}{j}}{\binom{n}{j}^2} \\
&= k\frac{(t+1)\cdot(n-t)}{n^2} + \alpha_{n,t,k} \text{ with } 0 \leq \alpha_{n,t,k} \leq 2^t
\end{aligned}$$

We can thus rewrite τ's expectation as following.

$$
\begin{aligned}
E(\tau) &= \sum_{t=0}^{n-1} \frac{1}{p_t} \\
&= \sum_{t=0}^{n-1} \frac{n^2}{k \cdot (t+1) \cdot (n-t) + \frac{\alpha_{n,t,k}}{n^2}} \;<\; \sum_{t=0}^{n-1} \frac{n^2}{k \cdot (t+1) \cdot (n-t)} \\
&< \frac{1}{k} \cdot \frac{n^2}{n+1} \cdot \sum_{t=0}^{n-1} \left(\frac{1}{t+1} + \frac{1}{n-t} \right) \\
&< \frac{2}{k} \cdot \frac{n^2}{n+1} \cdot \sum_{t=1}^{n} \left(\frac{1}{t} \right) \\
&< \frac{2}{k} \cdot \frac{n^2}{n+1} \cdot H_n \\
&< \frac{2}{k} \cdot \frac{n^2}{n+1} \cdot \left(\ln(n) + \gamma + \mathcal{O}(\frac{1}{2n}) \right) \text{ with } \gamma = \lim_{n\to\infty} (H_n - \ln(n))
\end{aligned}
$$

$$
\begin{aligned}
var(\tau) &= \sum_{t=0}^{n-1} \frac{1-p_t}{{p_t}^2} \;<\; \sum_{t=0}^{n-1} \frac{1}{{p_t}^2} \\
&< \sum_{t=0}^{n-1} \frac{1}{\left(k\frac{(t+1)\cdot(n-t)}{n^2} + \alpha_{n,t,k} \right)^2} \;<\; \sum_{t=0}^{n-1} \frac{1}{\left(k\frac{(t+1)\cdot(n-t)}{n^2} \right)^2} \\
&< \frac{n^4}{k^2} \left(\sum_{t=0}^{n/2-1} \frac{1}{((t+1)\cdot(n-t))^2} + \sum_{t=n/2}^{n-1} \frac{1}{((t+1)\cdot(n-t))^2} \right) \\
&< 2 \cdot \left(\frac{n}{k} \right)^2 \cdot \left(\frac{n}{n/2} \right)^2 \cdot \sum_{t=0}^{n/2-1} \frac{1}{(t+1)^2} \\
&< \frac{4}{3}\pi^2 \cdot \left(\frac{n}{k} \right)^2
\end{aligned}
$$

To now prove the lower bound of τ, we will compare the number of fixed points of a permutation σ, $F(\sigma)$, for our shuffle, the permutation obtained from the identity by applying kt random transpositions $P^{kt}(id, \cdot)$, and the uniform distribution π, or more precisely compare the corresponding probabilities over the set $A = \{\sigma : F(\sigma) \geq \frac{\mu}{2}\}$. We can say that after t shuffles, the number of untouched cards of our shuffle has the same distribution as the number R_{2kt} of uncollected coupon types after $2kt$ steps of a coupon collector chain and that about $P^{kt}(id, \cdot)$ that the associate $F(\sigma)$ is at least as large as the number of cards that were touched by none of the transpositions, i.e. $P^{kt}(id, A) \geq P(R_{kt} \geq A)$.

We know that the R_{2kt} has expectation $\mu = np$ with $p = \left(1 - \frac{1}{n}\right)^{2kt}$, variance $var = np(1-p) < \mu$ and by Chebyshev, we know that $\Pr(R_{2kt} \leq \frac{\mu}{2}) \leq \frac{4}{\mu}$ as $\Pr(|R_{2kt} - \mu| \geq \frac{\mu}{2}) = \Pr(R_{2kt} \geq \frac{3\mu}{2}) + \Pr(R_{2kt} \leq \frac{\mu}{2}) > \Pr(R_{2kt} \leq \frac{\mu}{2})$.

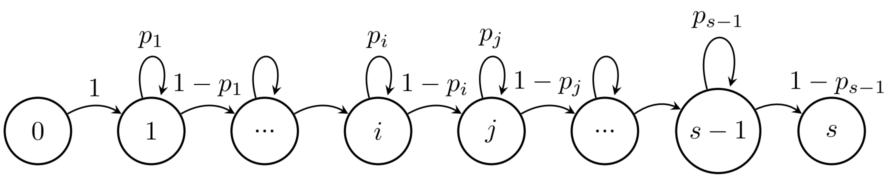

Figure 1: Access Markov Chain, with $p_i = \frac{i}{n}$

$$\begin{bmatrix} 0 & 1 & & & & \text{\Large 0} \\ 0 & p_1 & 1-p_1 & & & \\ & & p_2 & 1-p_2 & & \\ & & & 1-p_3 & p_3 & \\ & & & & p_4 & 1-p_4 \\ & \text{\Large 0} & & & & 1 \end{bmatrix}$$

Figure 2: Corresponding Transition Matrix P, for $s = 5$

By Markov's inequality we know that $\pi(A) \leq \frac{2}{\mu}$.

As $P^{kt}(id, A) \geq P(R_{kt} \geq A)$, we also have $P^{kt}(id, A^c) \leq P(R_{2kt} \leq A) \leq \frac{4}{\mu}$ which leads to $P^{kt}(id, A) \geq 1 - \frac{4}{\mu}$.

Thus we have $d(t) = ||P^{kt}(id,) - \pi||_{TV} \geq |1 - \frac{4}{\mu} - \frac{2}{\mu}| \geq 1 - \frac{6}{\mu}$.

We want to find the minimum t such that $1 - \frac{6}{\mu} \geq \epsilon$, which is equivalent to $n \cdot \left(1 - \frac{1}{n}\right)^{2kt} \geq \frac{6}{1-\epsilon}$ and to

$$\log\left(\frac{n \cdot (1-\epsilon)}{6}\right) \geq 2kt \log\left(\frac{n}{n-1}\right)$$

As $\log(1+x) \leq x$, the previous inequality holds if $\log\left(\frac{n\cdot(1-\epsilon)}{6}\right) \geq \frac{2kt}{n-1}$ which means that if $t \leq \frac{n-1}{2k} \cdot \log\left(\frac{n(1-\epsilon)}{6}\right)$ then $d(t) \geq \epsilon$. Thus,

$$\tau(\epsilon) \geq \frac{n-1}{2k} \ln(n \cdot \frac{1-\epsilon}{6})$$

. □

A.1.2 Proof of fake accesses

Proof. We want to prove that the average number of fake access is 0 in case of a uniform distribution. To do so, we consider the markov chain in Fig 1 and its Transition Matrix as depicted in Fig 2 for $s = 5$. The transition matrix P represents the s transient state, in which the stash is not completely filled, and the absorption state in which the stash is full. Thus, P can be decomposed in 4 submatrices: the square submatrix Q_s representing all the transcient state, the column matrix R with the probabilities of transitioning to the absorbing state, the null row matrix and the absorption matrix.

$$\begin{bmatrix} Q_s & R \\ 0_{1\times s} & I_1 \end{bmatrix}$$

To find the average number of steps from one state to the absorbing one, we have solve the following equation, each row corresponding to the average number of steps from the corresponding state (the stashed filled with some records) to the state where the stash is full.

$$\begin{aligned} t &= \left(\sum_{k=0}^{\infty} {Q_s}^k\right) 1 \\ &= (I_s - Q_s)^{-1} 1 \end{aligned}$$

This equation has a solution since $M = I_s - Q_s$ have independant rows and thus an inverse that we call N. By calculus we find that,

$$\begin{aligned} n_{i,j} &= 0 && \text{if } i > j, \\ n_{i,j} &= \frac{1}{m_{i,i}} && \text{if } i = j, \\ n_{i,j} &= -n_{i+1,j} \cdot \frac{m_{i,i+1}}{m_{i,i}} && \text{if } i < j \end{aligned}$$

which can be simplified by

$$\begin{aligned} n_{i,j} &= 0 && \text{if } i > j, \\ n_{i,j} &= \frac{1}{m_{j,j}} \prod_{k=1}^{j-1} \left(-\frac{m_{i,k+1}}{m_{k,k}}\right) && \text{if } i \leq j \end{aligned}$$

We only want to calculate the first solution S_1 from the equation.

$$\begin{aligned} S_1 &= \sum_{j=0}^{s-1} \frac{1}{m_{j,j}} \prod_{k=1}^{j-1} \left(-\frac{m_{i,k+1}}{m_{k,k}}\right) \\ &= \sum_{j=1}^{s-1} \frac{1}{m_{j,j}} \text{ as } m_{i,k+1} = -m_{k,k} \\ &= \sum_{j=0}^{s-1} \frac{1}{1-\frac{j}{n}} \\ &= \sum_{j=0}^{s-1} \frac{n}{n-j} \\ &= \sum_{j=0}^{s-1} \left(1 + \frac{j}{n-j}\right) \\ &= s + \sum_{j=0}^{s-1} \frac{j}{n-j} \end{aligned}$$

As s steps are required to fill the stash, we thus find the following inequality for the number of fake access f:

$$\frac{s \cdot (s+1)}{2 \cdot n} < f < \frac{s \cdot (s+1)}{2 \cdot (n+1-s)}$$

$\square$