Lightweight Authentication of Linear Algebraic Queries on Data Streams

Stavros Papadopoulos (HKUST), Graham Cormode (University of Warwick)
Antonios Deligiannakis (Technical University of Crete), Minos Garofalakis (Technical University of Crete)

Problem Definition

Motivation: A company may not possess the resources for deploying a DSMS.
Solution: The company outsources its data stream storage and management to a third-party server.
Challenge: The server may be untrustworthy: result integrity and freshness must be ensured to the clients.

Result Summary: For 3 important functions (vector sum, dot product, matrix product) we show secure and lightweight schemes that allow the client to check the computation of the server.

Architecture

Solution

Motivation: A company may not possess the resources for deploying a DSMS.
Solution: The company outsources its data stream storage and management to a third-party server.
Challenge: The server may be untrustworthy: result integrity and freshness must be ensured to the clients.

Result Summary: For 3 important functions (vector sum, dot product, matrix product) we show secure and lightweight schemes that allow the client to check the computation of the server.

Dynamic Vector Sum (DVS)

Setting
- There are $m$ machines generating $m$ streams.
- Stream $X_i$ updates an $n$-element vector $a_i$ at $M_i$.
- The query result is $\sum_{i=1}^{m} a_i$.

Our Results:
- $O(1)$ costs at $M_i$.
- $O(m)$ processing cost and $O(1)$ space at the server.
- $O(m+n)$ verification cost at the client.
- $O(1)$ proof size (a few bytes).
- All operations are lightweight (order of a few $\mu$s).

Dynamic Matrix Product (DMP)

Setting
- Machines $M_a, M_b$ generate streams $X_a, X_b$, respectively.
- $X_a(X_b)$ updates an $n_a \times n$ ($n \times n_b$) matrix $A \cdot B$.
- The query result is $n_a \times n_b$ matrix $A \cdot B$.

Our Results:
- $O(1)$ update and $O(n)$ space/comm. cost at $M_a, M_b$.
- $O(n)$ processing cost and $O(1)$ space at the server.
- $O(n_a n_b)$ verification cost at the client.
- $O(1)$ proof size.
- All operations are lightweight.

Dynamic Dot Product (DDP)

Setting
- Machines $M_a, M_b$ generate streams $X_a, X_b$, respectively.
- $X_a(X_b)$ updates an $n$-element vector $a \cdot b$.
- The query result is the dot product $a \cdot b$.

Our Results:
- $O(1)$ costs at $M_a, M_b$.
- $O(n \log n)$ process. and $O(n)$ space at the server.
- $O(1)$ verification cost at the client.
- $O(1)$ proof size.
- All operations at the client and $M_a, M_b$ are lightweight (the server requires exponentiations).

Solution idea:
- $M_i$ incrementally maintains summary $S_i = \sum_{j=1}^{k_i} a[i][j]$ in a finite field, where $k_i$ are secret keys.
- $M_i$ signs $S_i$ with a variant of one-time pad encryption.
- All keys are produced from $sk$.
- The server computes proof $\pi_i = \sum_{j=1}^{k_i} \sigma_i[j]$.
- The client can verify $\pi_i$ with the result and $sk$.
- Security is based on the security of pseudorandom functions (PRFs).

Applications:
- Group by queries (e.g., for network analysis).
- Sum and count queries in sensor networks.

Summary

- Event co-occurrence in monitoring applications.
- Joint frequency distribution of attributes in joins.

Solution idea:
The matrix product is the summation of outer products between a column from $A$ and a row from $B$.
- $M_a$ ($M_b$) maintains summary $S_a[j]$ ($S_b[i]$) for each $n$-element column (row) $j$, similar to DVS.
- $\sum_{i=1}^{m} S_a[j] \cdot S_b[i]$ is an (unsigned) summary for $A \cdot B$.
- A trick is needed to handle the one-time pad nonces.
- Security is based on the security of PRFs.

Applications:
- Event co-occurrence in monitoring applications.
- Joint frequency distribution of attributes in joins.

Solution idea:
The result is the dot product $a \cdot b$.
- Create a signed summary for $a \otimes b$ similar to DMP, and assist the server to remove unnecessary terms.
- To avoid giving key material to the server, we provide (offline and only once as public info $pub$) the key information in the exponent of a group generator—all computations move to the exponent.
- Security is based on the security of PRFs and the Diffie-Hellman Exponent (n-DHE) assumption.

Applications:
- Joins.
- Similarity queries.