Secure Information Processing with Privacy Assurance (SIPPA)

Bon K. Sy, Adam Ramirez*, Arun Prakash Kumara
Queens College of The City University of New York
What to expect

Example Scenario

The inner mechanisms of SIPPA

Experimental results
The Scenario
Information Swap

A user wants to send their biometric fingerprint data to a server, only if the fingerprint is similar to the template on the server.
The Conditions

SIPPA meets the following 2 conditions:

1. Neither of the two parties (client and server) involved in SIPPA share their private data with each other.

2. If the data is "sufficiently similar", the server party will provide the client party with helper data so that the client may reconstruct the server data.
Inner Workings of SIPPA
1: Derive symmetric matrix representation of the both client and server data

2: Of these calculated matrices we must now derive the eigenvalues and the corresponding unity normalized eigenvectors
How SIPPA works

Relationship between Matrices & Eigenvectors

Deviance of M1 and M2

Approx. Linear Relationship

Deviance of V1 and V2

\[ V_i = \text{Eigenvector of } M_i \]

\[ i = 1, 2 \]
Example

\[ M_1 = \begin{bmatrix} 200 & 150 \\ 55 & 255 \end{bmatrix}, \quad D_1 = \begin{bmatrix} 200 \\ 150 \\ 55 \\ 255 \end{bmatrix} \]

\[ A_1 = \begin{bmatrix} 40000 & 30000 & 11000 & 51000 \\ 30000 & 22500 & 8250 & 38250 \\ 11000 & 3250 & 3025 & 14025 \\ 51000 & 38250 & 14025 & 65025 \end{bmatrix} \]

\[ \lambda_1 = 130550, \quad V_1 = \{40,30,11,51\} \]
3: Compute a vector $x$ such that when transformed by matrix $(A1+A2)$ it results in the sum of both client and server eigenvalues and corresponding eigenvectors.
Conditions Within SIPPAA

Party 1 (Private Data)
Matrix Data ($A_1$)
Largest Eigenvalue ($\lambda_1$)
Eigenvector ($V_1$)

A1V1 = $\lambda_1 \cdot V_1$

Party 2 (Private Data)
Matrix Data ($A_2$)
Largest Eigenvalue ($\lambda_2$)
Eigenvector ($V_2$)

A2V2 = $\lambda_2 \cdot V_2$

($A_1 + A_2$) x = $\lambda_1 \cdot V_1 + \lambda_2 \cdot V_2$
How \( x \) is calculated

Using PPCSC

\[
(A_1 + A_2) \cdot x = \lambda_1 \cdot V_1 + \lambda_2 \cdot V_2
\]

\[
P_1 (A_1 + A_2) P_2 y = P_1 (\lambda_1 \cdot V_1 + \lambda_2 \cdot V_2)
\]

Client: \((P_1, A_1, \lambda_1 \cdot V_1)\)
Server: \((P_2, A_2, \lambda_2 \cdot V_2)\)

P1 & P2 are random invertible matrices

After SVD:
Once \( y \) is solved for, the server can now solve \( x = P_2 y \)

\( x \) is then sent from the server to the client

If \( x \) is intercepted, it will not be possible to reconstruct client or server data unless the interceptor has either \((A_1, P_1, V_2)\) or \((A_2, P_2, V_1)\)
SIPPA Algorithm Summary

4: Derive the closeness of the client and server data sets based on the minimum distance of their eigenvectors.
A1 & A2 are the same or have very low deviance
\[(A1 + A2) \times = \lambda_1 \cdot V1 + \lambda_2 \cdot V2\]

\[A1 \cdot x + A2 \cdot x = A1 \cdot V1 + A2 \cdot V2\]

A1 & A2 are the same or have very low deviance
\[(A1 + A2) \cdot x = \lambda_1 \cdot V1 + \lambda_2 \cdot V2\]

\[A1 \cdot x + A2 \cdot x = A1 \cdot V1 + A2 \cdot V2\]

A1 & A2 are not the same and have high deviance
SIPPA Algorithm Summary

5: If the client and server data are sufficiently close in comparison to some pre-defined threshold, the server will send the client helper data.

6: Lastly, the client will estimate the server’s eigenvector and use that estimate to estimate the entire matrix of the server resulting in an estimate of the server side data.
Reconstruction

**Given:**

Client: $A_1, \lambda_1, V_1, x$

Server: $A_2, \lambda_2, V_2, x$

**Helper Data:**

\[
\lambda_2, (\text{De})^T x^{-1}
\]

In the equation

\[
A_1x + A_2x = \lambda_1 \cdot V_1 + \lambda_2 \cdot V_2
\]

, $A_2x \approx \lambda_2 V_2$

Since $A_2 = \text{De} \cdot \text{De}_T$, the client can now calculate De by first calculating $V_2$

**Solve for $V_2$:**

\[
V_2' = V_1 + 2(x - V_1)
\]

\[
\text{De} \cdot \approx \frac{V_2' \cdot \lambda_2}{(\text{De})^T x}
\]
Vector Sizes

A normal 640x480 image would result in a 307200 x 307200 matrix.

We split the original matrix into sub-matrices and linearized them into the proper sub-vectors.

How do we know how small to make the sub-vectors?
Experimental Results
Design of Experiments

Random generation of 5 data sets categorized by dimension

Dimensions of 5, 10, 20, 30, 60 with 10 pairs of data each.

Key Questions
1. How close does SIPPA estimate based on deviance of the initial matrices?
2. How is the estimate affected by the dimension of the submatrix?
3. How is the estimate affected by the directional deviation of V1 and V2?
Experimental Results

Determine optimal settings for calculation on PGM file
Verify Linear Relationship

A linear relationship must be revealed between the sub-matrices and the eigenvalue/eigenvectors of the sub-matrices.
Estimation Power of SIPPA

The deviance between sample and template images as well as sub-vector size affect the estimation power of SIPPA.
Trial 1

SIPPA using real images (iterative)

Source Image

Similar Image
Reconstruction using similar image

Reconstruction using dissimilar image
Blue circles represent key differences between two regenerated images
Complexity of SIPPAA

In all cases, \( n = \) column size of Matrix

Matrix Composition \( O(n^2) \)

Eigenvalue/Eigenvector Calculation \( O(n^3) \)

Singular Value Decomposition \( O(n^2r) \) \( r = \) rank of matrix

Euclidean Distance Calculation \( O(n^2) \)

Overall Complexity of One SIPPAA Iteration - \( O(n^3) \)
Trial 2 - ParallelSIPPA  Parallel Design with BioAPI based implementation
Parallel Design
PSIPPA Results

Total of 10 different experiments

Lowest times for 256x192 and 640x480, 2min and 15min respectively

<table>
<thead>
<tr>
<th>#</th>
<th>Machine Setup</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Two DC’s (Dual Core-3.2GHz, 2.75GB RAM)</td>
</tr>
<tr>
<td>2</td>
<td>Two VM-512’s (Virtual Machine-1GHz, 5G RAM)</td>
</tr>
<tr>
<td>3</td>
<td>One QC. (Quad Core Machine-3.2GHz, 4G RAM)</td>
</tr>
<tr>
<td>4</td>
<td>Eight DC’s Running multiple SS(SIPPA-Server) Instances and Eight DC’s Running multiple SC(SIPPA-Client) instances.</td>
</tr>
<tr>
<td>5</td>
<td>Five VM-512’s Running multiple SS Instances and Five VM-512’s Running multiple SC instances.</td>
</tr>
<tr>
<td>6</td>
<td>16 (2 used for SS and 14 used for SC) DC’s Running Multiple Instances Of SS and SC</td>
</tr>
<tr>
<td>7</td>
<td>8 (1 used for SS and 7 used for SC) DC’s Running Multiple Instances Of SS and SC</td>
</tr>
<tr>
<td>8</td>
<td>10 (2 used for SS and 8 used for SC) VM-512’s Running Multiple Instances Of SS and SC</td>
</tr>
<tr>
<td>9</td>
<td>10 (2 used for SS and 8 used for SC) VM-1G’s Running Multiple Instances Of SS and SC</td>
</tr>
<tr>
<td>10</td>
<td>5 (1 used for SS and 4 used for SC) VM-1G’s Running Multiple Instances Of SS and SC</td>
</tr>
</tbody>
</table>

Engaging in SIPPA iteratively i.e. processing one 10X1 vector at a time.
Optimized Parallel SIPPA where all CPU’s are maintained at 100% usage for most of the time.
Naive Parallel SIPPA, Equal number of Physical Machines For SS & SC.
Questions?
Thank you

Reference: