PRIVACY-PRESERVING MUSICAL DATABASE MATCHING
Madhusudana Shashanka, Paris Smaragdis
--------------------------------------------

ABSTRACT
Match a piece of music audio to a service database using a privacy preserving process for both sides

INTRODUCTION
Secure Multi-party Communication (SMC) protocols
- allow multiple parties to perform arbitrary collaborative computations while guaranteeing the privacy of their data
- originated from the millionaire problem
- machine learning community: performing k-means, computation of means and related statistics from distributed databases, computer vision applications

PROBLEM FORMULATION
Solution with no privacy issues: Alice sends a piece of audio information $x$ to Bob, Bob computes the cross-correlation $c_k$ between $x$ and all the songs $y_k$ in his database, Bob sends all the cross-correlations to Alice, Alice gets the index $k$ of the maximum value of all the $c_k$, Alice accesses the database using $k$ to find her song information

BACKGROUND
Secure two-party computation
A protocol that implements an algorithm to calculate $c=f(a,b)$ is said to be secure only if it leaks no more information about $a$ and $b$ than what one could gain from learning $c$ from a trusted third-party.

General strategy to create a secure version of the algorithm
- express every step in terms of basic operations for which secure implementations are already known
- distribute intermediate results randomly between the two parties (e.g. random additive shares $z_1$ and $z_2$ such that $z=z_1+z_2$ is the intermediate result)

Homomorphic public-key cryptosystem
A public-key cryptosystem is a set of probabilistic polynomial-time algorithms for key generation (sk,pk), encryption (pk) and decryption (sk); pk: public key, sk: private key.
A cryptosystem is called homomorphic if one can indirectly perform specific algebraic operations on the encrypted data by manipulating the cyphertext. In this case, $EN(a,pk) * EN(b,pk) = EN(a+b,pk)$.
SECURE MUSIC COMPARISON

**Step 1: Cross-correlating the music signals**
Use the fact that cross-correlation is a sliding inner-product

Alice: \( GE(pk, sk) \rightarrow e_t = EN(x_t, pk), \ t = 1, \ldots, T \)
send \( pk \) and all \( e_t \) to Bob

Bob: \( z_n = \prod_t e_t \cdot y_k \cdot EN(c_k, pk), \ z_n' = z_n \cdot EN(-b_k, pk), \ b_k \) random
send all \( z_n' \) to Alice

Alice: \( ak_n = DE(z_n', sk) = c_k_n - b_k_n \rightarrow \) random additive shares \( c_k = ak + bk \)

**Step 2: Obtaining the cross-correlation peaks**
\( c_k = ak + bk \rightarrow c_k \succeq c_j \Leftrightarrow (ak_i - ak_j) \geq (bk_i - bk_j) \)

Yao's solution to the millionaire problem:
Alice has \( i \), Bob has \( j \), \( 1 < i, j < 10 \), decide whether \( i < j \)
M: set of all \( N \)-bit integers, QN: set of all 1-1 onto functions \( M \rightarrow M \)

Alice: \( GE(pk, sk) \), \( pk \) is a random element of QN
send \( pk \) to Bob

Bob: \( k = EN(x, pk) \), \( x \) is a random \( N \)-bit integer
send \( k - j + 1 \) to Alice

Alice: \( y_u = DE(k - j + u, sp), u = 1, \ldots, 10 \)
\( z_u = y_u \mod p \), \( p \) is a random \( N/2 \)-bit prime
send \( p \) and \( z_1, \ldots, z_i, z_{i+1}, \ldots, z_{10} \) (10 numbers) to Bob

Bob: if \( j^\text{th}_z = x \mod p \), then \( i \geq j \); otherwise \( i < j \) \footnote{\( j^\text{th}_z = x + DE(-1, sk) + 1 \)}
send result to Alice

**Step 3: Finding the most likely song index**
Permute protocol: \( A' + B' = \Pi(A') + \Pi(B') \)

Bob: \( GE(pk, sk) \rightarrow EN(B_i, pk), i = 1, \ldots, K \)
send \( pk \) and all \( EN(B_i, pk) \) to Alice

Alice: \( \theta_i = EN(B_i, pk) \cdot EN(S_i, pk) = EN(B_i + S_i, pk), S_i \) random
\( \theta_i \rightarrow \pi(\theta_i) \), permutation (i.e. song index scrambling)
send all \( \pi(\theta_i) \) to Bob

Bob: \( B_i' = DE(\pi(\theta_i), sk) = \Pi(B_i + S_i) \)

Alice: \( A_i' = \Pi(A_i - S_i) \)

**Final Step: Obtaining the desired information**
Oblivious transfer: used in situations where the database is private

Alice: \( GE(pk, sk) \rightarrow z = EN(\sigma, pk), \ \sigma \) is the index of the song
send \( z \) to Bob

Bob: \( dk = EN(uk, pk) \ast (z \ast EN(-k, pk))^r_k, uk \) is the encoded catalog info
\( r_k \) random, \( k = 1, \ldots, K \)
send all \( dk = EN(uk + r_k(\sigma - k), pk) \) to Alice

Alice: \( DE(dk) = uk + r_k(\sigma - k) \)
use the \( \sigma \)-th value to obtain the desired information \( u_\sigma = DE(d\sigma) \)