Phasing: Private Set Intersection using Permutation-based Hashing

Michael Zohner (TU Darmstadt)

Joint work with
Benny Pinkas (Bar Ilan University)
Thomas Schneider (TU Darmstadt)
Gil Segev (Hebrew University)
Private Set Intersection (PSI)

\[ X \cap Y \]
Application: Common Contacts
Application: Online Advertisement
Additional Applications

Secure database join

Botnet detection

Cheater detection in online games

Testing human genomes

Relationship path discovery
A naïve but insecure PSI protocol

**Input:** $x_1, \ldots, x_n$

$H(x_1), \ldots, H(x_n)$

**Input:** $y_1, \ldots, y_n$

$H(y_1), \ldots, H(y_n)$

$H(x_i) \neq H(y_j)$, for $0 < i, j < n$

**Pro:** fast, little communication

**Con:** insecure, can leak privacy of Bob's inputs
PSI Classification [PSZ14]

Public-key Cryptography

Generic Secure Computation

Oblivious Transfer

This talk: semi-honest (passive) adversaries
Public-key Cryptography

Protocols have existed for three decades

Encrypt elements using public-key crypto

Protocols based on public-key cryptography

- DH-based Protocol [M86], $O(n)$ pk-crypto & comm
- Blind RSA Protocol [CT10], $O(n)$ pk-crypto & comm
Generic Secure Computation techniques represent a function as Boolean circuit and operate on single bits.

Techniques are Yao's garbled circuits and GMW.

The sort-compare-shuffle circuit [HEK12] for PSI requires $O(n\sigma \log n)$ sym-crypto & comm, for element bit-length $\sigma$ and set size $n$. 


Oblivious Transfer (OT)

Input: Bob holds two strings \((s_0, s_1)\), Alice holds a choice bit \(c\)

Output: Alice only learns \(s_c\); Bob learns nothing about \(c\)

OT-based PSI protocols for sec. param. \(\kappa; \sigma\) bit elements:
Bloom-filter [DCW13], \(O(n\kappa)\) sym-crypto, \(O(n\kappa^2)\) comm
OT+Hashing [PSZ14], \(O(n\sigma)\) sym-crypto & comm
Performance Classification [PSZ14]

PSI on $n = 2^{18}$ elements of $\sigma=32$-bit length for 128-bit security on Gbit LAN

PK-Based:
- high run-time for large security parameters
  + best communication

Circuit-Based:
- high run-time & communication
  + easily extensible to arbitrary functions

OT-Based:
+ good communication and run-time

OT-Based:
+ good communication and run-time

PK-Based:
- high run-time for large security parameters
  + best communication

Circuit-Based:
- high run-time & communication
  + easily extensible to arbitrary functions

OT-Based:
+ good communication and run-time

OT-Based:
+ good communication and run-time
Our Contributions

**Goal:** Make PSI protocols more practical

**Phasing:**
PSI using Permutation-based Hashing

**Circuit-Phasing:**
Improvements on Circuit-based PSI [HEK12]

**OT-Phasing:**
Improvements on OT+Hashing [PSZ14]
**OT+Hashing PSI [PSZ14]**

**Input:** Alice has $x$, Bob has $y$. **Output:** $x = y$

**Example:** $x = 001$, $y = 011$, $\sigma = 3$, stat. sec. param. $\lambda$

Bob sends $\lambda$-bit mask $0 \oplus 1 \oplus 1$ to Alice

Alice computes $0 \oplus 0 \oplus 1$ and compares
OT+Hashing PSI [PSZ14] (cont.)

Private Equality Test:

Private Set Inclusion:

Private Set Intersection:

\[ O(n^2) \text{ comparisons!} \]
Hashing to Bins [PSZ14]

Hash elements to bins to reduce comparisons

**Example:** Alice holds $X = \{x_1, x_2, x_3\}$, Bob holds $Y = \{y_1, y_2, y_3\}$

$$H(x_3) \rightarrow \begin{array}{c} \hline \end{array} x_3 \begin{array}{c} \hline \end{array} \rightarrow y_1 \rightarrow y_3$$

$$H(x_1) \rightarrow \begin{array}{c} \hline \end{array} x_1 \begin{array}{c} \hline \end{array} \rightarrow \begin{array}{c} \hline \end{array} \rightarrow \begin{array}{c} \hline \end{array}$$

$$H(x_2) \rightarrow \begin{array}{c} \hline \end{array} x_2 \begin{array}{c} \hline \end{array} \rightarrow \begin{array}{c} \hline \end{array} \rightarrow \begin{array}{c} \hline \end{array} \rightarrow \begin{array}{c} \hline \end{array}$$

Reduces comparisons from $O(n^2)$ to $O(n \log n / \log \log n)$
Our Contributions (1)

Phasing:
PSI using Permutation-based Hashing
Permutation-based Hashing

In [PSZ14] elements are **compared bit-wise**
- Hence, smaller elements require less overhead

Idea: “hash” elements to a **smaller representation**
- To avoid collisions the birthday paradox states that the hash must be \( \lambda + 2\log(n) \) bit

Instead: use a permutation to map elements to bins and store a shorter representation
- Used for smaller hash tables [ANS10]
- Here: first use in crypto
Split $x = x_L | x_R$ with $|x_L| = O(\log n)$ bit

Let $f: [1...2^{|x_R|}] \rightarrow [1...2^{|x_L|}]$ and $p(x) = x_L \oplus f(x_R)$

Hashing is done by storing $x_R$ in bin $p(x)$

Securely compare $x_R$ which is only $\sigma - |x_L|$ bit long
- Less complexity for comparison
- Larger sets mean less complexity for comparison
Our Contributions (2)

Circuit-Phasing
Circuit-Phasing

Idea: Use permutation-based hashing to hash elements into bins and compare bins on elements with reduced length.

For each bin compare the element of Alice with each element in the same bin of Bob using bit-wise comparison circuit.

Advantages:
- Communication rounds independent of set sizes
- Same circuit evaluated multiple time allows SIMD
However, bins have to be padded to avoid information leakage.

In total $O(n \log n / \log \log n)$ comparison circuits:

- Per comparison: $O(\sigma \log n)$ sym-crypto & comm
- Total: $O(n (\sigma \log n) \log n / \log \log n)$ sym-crypto & comm
- SCS circuit [HEK12]: $3n\sigma \log n$ sym-crypto & comm
Improvements Circuit-based PSI

PSI on \( n = 65,000 \) elements of \( \sigma = 32 \)-bit length for 128-bit security on Gbit LAN
Our Contributions (3)

OT-Phasing
OT-Phasing

Use permutation hashing in OT+Hasing protocol [PSZ14]

Further protocol optimizations:

Use more hash functions for the hashing-to-bins routine
  • decreases number of bins by factor 2

Generate only one random string per bin
  • decreases client's work for larger sets
Improvements OT-based PSI

PSI on varying set sizes of different length for 128-bit security on Gbit LAN

![Graph showing run-time vs number of elements]
PSI on $n=16$ mio elements of different length for 128-bit security on Gbit LAN

OT-Phasing ($\sigma=32$) vs. OT-based [PSZ14] ($\sigma=32$) vs. Naïve ($\sigma=32$) vs. DH-ECC [M86]
Conclusion

More efficient PSI protocols with reduced overhead
  • Only factor 3 slower than currently used (insecure) solutions

Permutation hashing to reduce bit-length of elements

More efficient and scalable Circuit-based PSI

Code is online on GitHub http://encrypted.de/PSI
Questions?
References


