Google | Privacy & Data Protection Office

PIR-with-Default and Applications

PRIVATE COMPUTING

Tancrède Lepoint*

Sarvar Patel
Google

Mariana Raykova
Google

Karn Seth
Google

Ni Trieu*
Arizona State

*Work done while at Google

Problem Statement

“Inner-Join” Private Join and Compute

“Inner-Join” Private Join and Compute

User

X	**V**
x_1	v_1
...	...
x_m	v_m

ID

Weight

Google

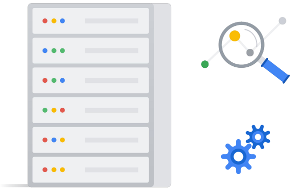
“Inner-Join” Private Join and Compute

User

X	V
x_1	v_1
...	...
x_m	v_m

ID — Weight

Server



Y	W
y_1	w_1
...	...
...	...
...	...
...	...
y_n	w_n

ID — Weight

“Inner-Join” Private Join and Compute

$$\sum_{x \in X \cap Y} V[x] \cdot W[x]$$

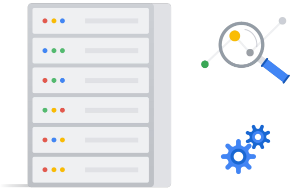
User

X	V
x_1	v_1
...	...
x_m	v_m

ID

Weight

Server



Y	W
y_1	w_1
...	...
...	...
...	...
...	...
y_n	w_n

ID

Weight

"Inner-Join" Private Join and Compute

$$\sum_{x \in X \cap Y} V[x] \cdot W[x] + \varepsilon$$

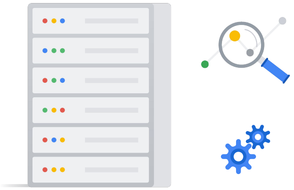
User

X	V
x_1	v_1
…	…
x_m	v_m

ID

Weight

Server



Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

ID

Weight

“Inner-Join” Private Join and Compute

$$\sum_{x \in X \cap Y} V[x] \cdot W[x] + \varepsilon$$

User

X	V
x_1	v_1
...	...
x_m	v_m

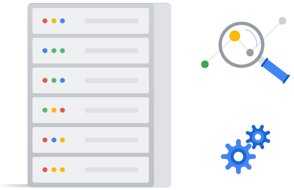
ID — Weight

Nothing more should be learned

Y	W
y_1	w_1
...	...
...	...
...	...
...	...
y_n	w_n

ID — Weight

Server



Google

Functionality/ Efficiency

- User should learn the dot product of weights (perhaps with noise added) for IDs in the intersection $X \cap Y$.
- User's communication and computation cost should be $\tilde{O}(|X|)$.
    - "Almost linear" in the User's data size.
    - Should grow very slowly with the Server's data size.
    - Assumption is that $|X| << |Y|$.

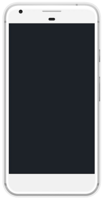
Privacy

- Parties' inputs should remain hidden.
- Elements of $X \cap Y$ should remain hidden. (Which IDs were in common)
- $|X \cap Y|$ should remain hidden. (Number of IDs in common)
- $|X|$ and $|Y|$ are OK to reveal. (Only input sizes, can be mitigated by padding inputs)

Application 1: Exposure notification (hypothetical)

$$\sum_{x \in X \cap Y} V[x] \cdot W[x] + \varepsilon$$

User



X	V
x_1	v_1
…	…
x_m	v_m

BLE ID

Proximity Weight

Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

BLE ID

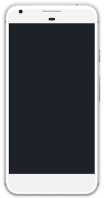
Virulence Weight

Health Authority

Application 1: Exposure notification (hypothetical)

$$\sum_{x \in X \cap Y} V[x] \cdot W[x] + \varepsilon$$

User



X	V
x_1	v_1
…	…
x_m	v_m

BLE ID

Proximity Weight

Health Authority

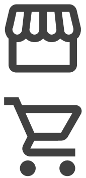
Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

BLE ID

Virulence Weight

Application 2: Measuring Ad effectiveness (hypothetical)

Merchant

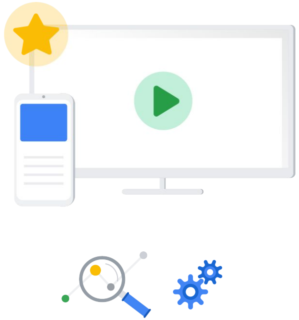


X	V
x_1	v_1
...	...
x_m	v_m

User ID

Spend Value

Ad Tech Company



Y	W
y_1	w_1
...	...
...	...
...	...
...	...
y_n	w_n

User ID

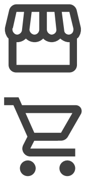
Time-decayed ad effect

Application 2: Measuring Ad effectiveness (hypothetical)

$$\sum_{x \in X \cap Y} V[x] \cdot W[x] + \varepsilon$$

"Weighted conversion credit"

Merchant



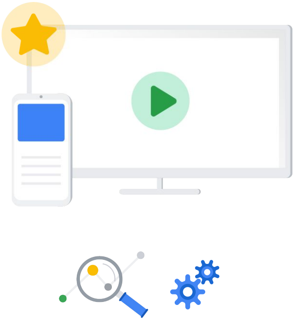
X	V
x_1	v_1
…	…
x_m	v_m

User ID

Spend Value

Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

Ad Tech Company

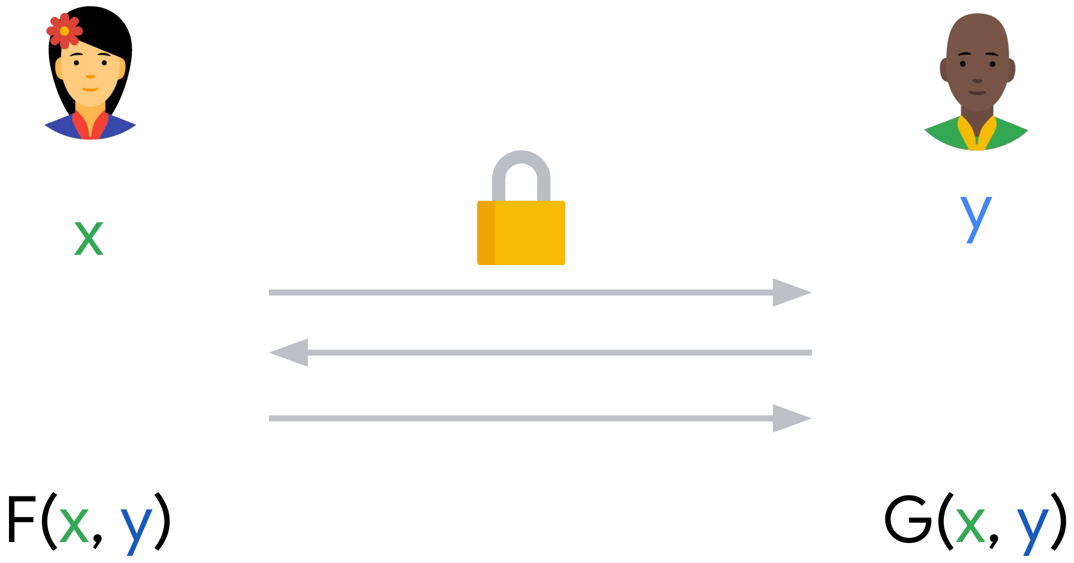


User ID

Time-decayed ad effect

Our Approach:
Secure Multiparty Computation

Secure Multiparty Computation (MPC)



Our Approach

Build a tailored MPC protocol for computing Inner Join PJC.

Our Approach

Build a tailored MPC protocol for computing Inner Join PJC.

Focusing on Asymmetric Input Sizes ($|Y| >> |X|$)

Desired Properties + Previous Work

Hides X ∩ Y				
Hides |X ∩ Y|				
Compute on Intersection				
User cost = Õ(|X|)				

Desired Properties + Previous Work

	Private Join and Compute[1]			
Hides $X \cap Y$	✓			
Hides $\lvert X \cap Y \rvert$				
Compute on Intersection	✓			
User cost = $\tilde{O}(\lvert X \rvert)$				

[1] Google Blog Post, "Helping organizations do more without collecting more data."

Desired Properties + Previous Work

	Private Join and Compute[1]	Private Information Retrieval[2]		
Hides $X \cap Y$	✓			
Hides $\|X \cap Y\|$				
Compute on Intersection	✓	✓		
User cost = $\tilde{O}(\|X\|)$		✓		

[1] Google Blog Post, "Helping organizations do more without collecting more data."
[2] https://en.wikipedia.org/wiki/Private_information_retrieval

Desired Properties + Previous Work

	Private Join and Compute[1]	Private Information Retrieval[2]	Circuit PSI[3]	
Hides $X \cap Y$	✓		✓	
Hides $\lvert X \cap Y \rvert$			✓	
Compute on Intersection	✓	✓	✓	
User cost = $\tilde{O}(\lvert X \rvert)$		✓		

[1] Google Blog Post, “Helping organizations do more without collecting more data.”
[2] https://en.wikipedia.org/wiki/Private_information_retrieval
[3] Pinkas, Schneider, Tkachenko, Yanai “Efficient Circuit-based PSI with Linear Communication”

Desired Properties + Previous Work

	Private Join and Compute[1]	Private Information Retrieval[2]	Circuit PSI[3]	Our Work
Hides $X \cap Y$	✓		✓	✓
Hides $\|X \cap Y\|$			✓	✓
Compute on Intersection	✓	✓	✓	✓
User cost = $\tilde{O}(\|X\|)$		✓		✓

[1] Google Blog Post, "Helping organizations do more without collecting more data."
[2] https://en.wikipedia.org/wiki/Private_information_retrieval
[3] Pinkas, Schneider, Tkachenko, Yanai "Efficient Circuit-based PSI with Linear Communication"

Desired Properties + Previous Work

*Also addressed by: Chen et al "Labeled PSI from Fully Homomorphic Encryption with Malicious Security"

	Private Join and Compute[1]	Private Information Retrieval[2]	Circuit PSI[3]	Our Work
Hides $X \cap Y$	✓		✓	✓
Hides $\|X \cap Y\|$			✓	✓
Compute on Intersection	✓	✓	✓	✓
User cost = $\tilde{O}(\|X\|)$		✓		✓

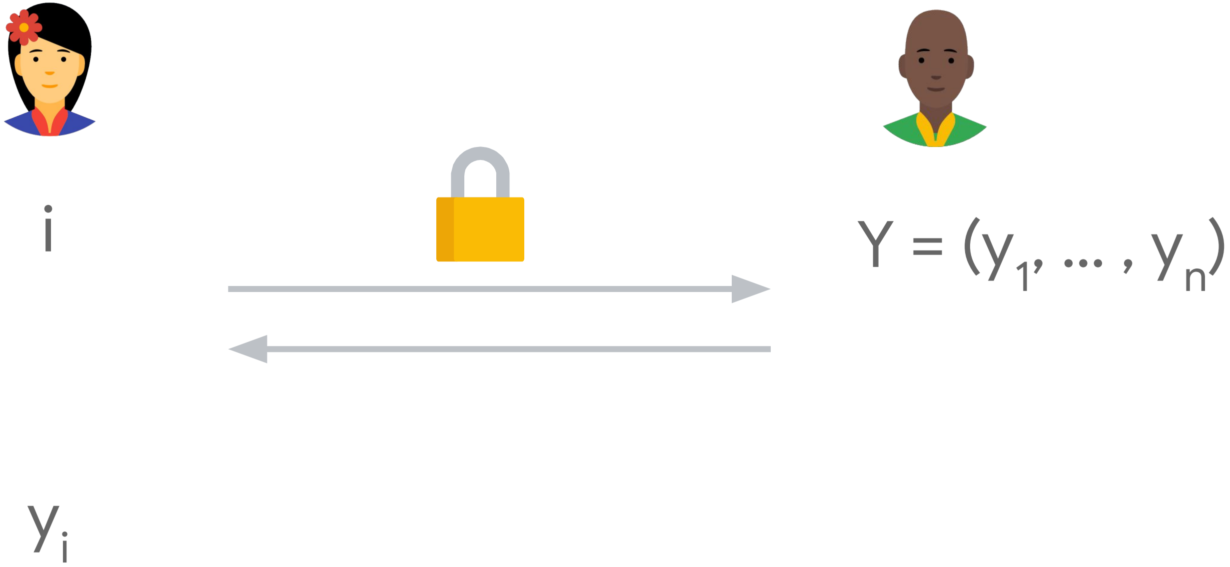
[1] Google Blog Post, "Helping organizations do more without collecting more data."
[2] https://en.wikipedia.org/wiki/Private_information_retrieval
[3] Pinkas, Schneider, Tkachenko, Yanai "Efficient Circuit-based PSI with Linear Communication"

Solution Overview

Starting Point: Private Information Retrieval (PIR)

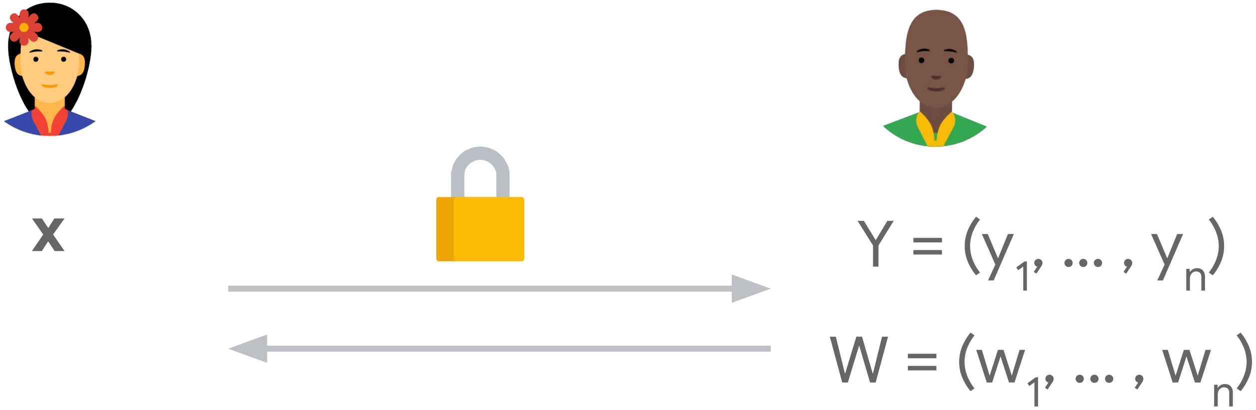
Google

Starting Point: Private Information Retrieval (PIR)



User Cost = $\tilde{O}(1)$

Step 1: Keyword PIR

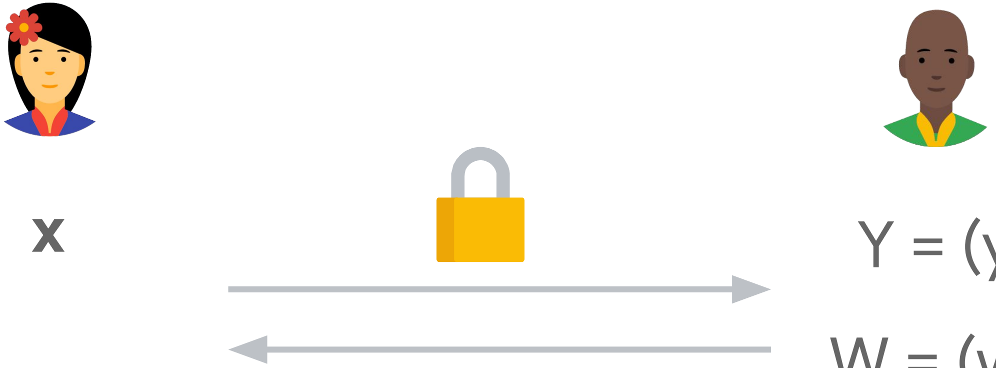


User Cost = $\tilde{O}(1)$

Step 1: Keyword PIR

User Cost = $\tilde{O}(1)$

Step 1: Keyword PIR



x

$Y = (y_1, \dots, y_n)$

$W = (w_1, \dots, w_n)$

W[x] or garbage

User Cost = $\tilde{O}(1)$

Step 2: PIR with Default

x

$Y = (y_1, \dots, y_n)$

$W = (w_1, \dots, w_n)$

t (default value)

W[x] or t

User Cost = $\tilde{O}(1)$

Step 3: PIR with Default + value

x, v

$Y = (y_1, \dots, y_n)$

$W = (w_1, \dots, w_n)$

t (default value)

(v * W[x]) or t

User Cost = Õ(1)

Step 3: PIR with Default + value + mask

x, v

$Y = (y_1, \ldots, y_n)$

$W = (w_1, \ldots, w_n)$

t (default value)

r (random mask)

$(v * W[x]) + r$ or $t + r$

User Cost = $\tilde{O}(1)$

Step 3: “**Extended**” PIR-with-Default

x, v

$Y = (y_1, \dots, y_n)$

$W = (w_1, \dots, w_n)$

t (default value)

r (random mask)

$(v * W[x]) + r$ or $t + r$

User Cost = $\tilde{O}(1)$

Putting it together: “Inner Join” PJC

Google

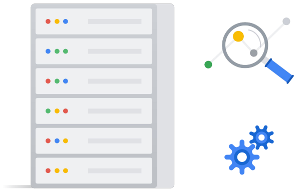
Putting it together: “Inner Join” PJC

User

X	V
x_1	v_1
…	…
x_m	v_m

Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

Server



Putting it together: “Inner Join” PJC

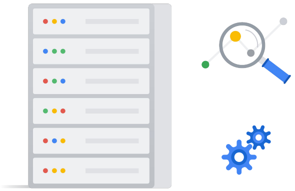
User

X	V
x_1	v_1
…	…
x_m	v_m

1. Execute Extended PIR-with-Default on each User input (x_i, v_i) with Server using default value 0 and a different random mask each time.

Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

Server



Putting it together: “Inner Join” PJC

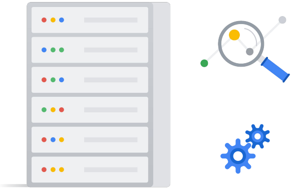
User

X	V
x_1	v_1
…	…
x_m	v_m

1. Execute Extended PIR-with-Default on each User input (x_i, v_i) with Server using default value 0 and a different random mask each time.
2. The user sums together the output it received from each execution to get a value T.

Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

Server



Putting it together: “Inner Join” PJC

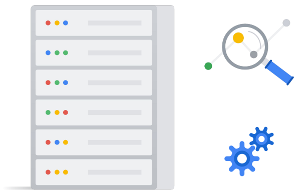
User

X	V
x_1	v_1
…	…
x_m	v_m

1. Execute Extended PIR-with-Default on each User input (x_i, v_i) with Server using default value 0 and a different random mask each time.
2. The user sums together the output it received from each execution to get a value T.
3. The server computes R, the sum of all random masks it used.

Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

Server



Putting it together: “Inner Join” PJC

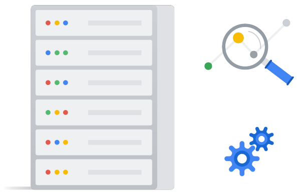
User

X	V
x_1	v_1
…	…
x_m	v_m

1. Execute Extended PIR-with-Default on each User input (x_i, v_i) with Server using default value 0 and a different random mask each time.
2. The user sums together the output it received from each execution to get a value T.
3. The server computes R, the sum of all random masks it used.
4. The server sends R’ = R - ε to the user for some noise ε.

Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

Server



Putting it together: “Inner Join” PJC

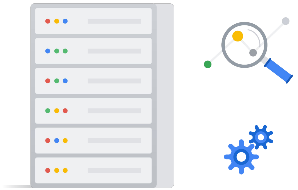
User

X	V
x_1	v_1
…	…
x_m	v_m

1. Execute Extended PIR-with-Default on each User input (x_i, v_i) with Server using default value 0 and a different random mask each time.
2. The user sums together the output it received from each execution to get a value T.
3. The server computes R, the sum of all random masks it used.
4. The server sends $R' = R - \varepsilon$ to the user for some noise ε.
5. The User outputs $T - R'$

Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

Server



Putting it together: “Inner Join” PJC

User

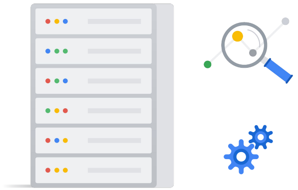
X	V
x_1	v_1
…	…
x_m	v_m

1. Execute Extended PIR-with-Default on each User input (x_i, v_i) with Server using default value 0 and a different random mask each time.
2. The user sums together the output it received from each execution to get a value T.
3. The server computes R, the sum of all random masks it used
4. The server sends R’ = R - ε to the user for some noise ε.
5. The User outputs T - R’

$$T - R' = \sum_{x \in X \cap Y} V[x] * W[x] + \varepsilon$$

Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

Server



PIR-with-Default construction

Starting Point: Private Information Retrieval (PIR)

Homomorphic encryption

Enc(i)

i

Starting Point: Private Information Retrieval (PIR)

i

Homomorphic
encryption

Enc(i)

Technically we encrypt a
special encoding of i, but
we elide the details

Starting Point: Private Information Retrieval (PIR)

i

Homomorphic encryption

Enc(i)

Expand using homomorphism

Enc(0)
...
...
...
Enc(1)
...
...
Enc(0)

Starting Point: Private Information Retrieval (PIR)

i

Homomorphic encryption

↓

Enc(i)

→

$Enc(0) * y_1$

... ...

... ...

...

$Enc(1) * y_i$

... ...

... ...

$Enc(0) * y_n$

Homomorphically Multiply

Starting Point: Private Information Retrieval (PIR)

i

Homomorphic encryption

Enc(i)

$$
\begin{array}{lcl}
\text{Enc}(0) & * & y_1 \\
\ldots & & \ldots \\
\ldots & & \ldots \\
\ldots & & \\
\text{Enc}(1) & * & y_i \\
\ldots & & \ldots \\
\ldots & & \ldots \\
\text{Enc}(0) & * & y_n \\
\hline
\text{Enc}(y_i) & &
\end{array}
$$

Homomorphically Sum

Starting Point: Private Information Retrieval (PIR)

Homomorphic encryption → Enc(i)

i → Enc(i)

$$
\begin{array}{lll}
\text{Enc}(0) & * & y_1 \\
\ldots & & \ldots \\
\ldots & & \ldots \\
\ldots & & \\
\text{Enc}(1) & * & y_i \\
\ldots & & \ldots \\
\ldots & & \ldots \\
\text{Enc}(0) & * & y_n \\
\hline
\text{Enc}(y_i) & &
\end{array}
$$

y_i ← Enc(y_i)

Google

Starting Point: Private Information Retrieval (PIR)

Homomorphic encryption

Enc(i)

Enc(0) * y_1

... ...

... ...

...

Enc(1) * y_i

... ...

... ...

Enc(0) * y_n

Enc(y_i)

i

What do we do if we have an ID/keyword instead of an index?

y_i

Bloom Filter (BF)

Bloom Filter (BF)

$$Y = (y_1, \dots, y_n)$$

$$\downarrow$$

$$BF_Y = (b_1, \dots, b_N)$$

Bloom Filter (BF)

x

$h_1(x), \ldots, h_k(x)$

$Y = (y_1, \ldots, y_n)$

$BF_Y = (b_1, \ldots, b_N)$

Bloom Filter (BF)

x

$h_1(x), \ldots, h_k(x)$

If $BF_Y[h_i(x)] = 1$
for all $i \in [k]$,
then  can conclude
that $x \in Y$
except with some failure
probability.

$Y = (y_1, \ldots, y_n)$

$BF_Y = (b_1, \ldots, b_N)$

Bloom Filter (BF)

x

$h_1(x), \ldots, h_k(x)$

$k = 31$

If $BF_Y[h_i(x)] = 1$
for all $i \in [k]$,
then  can conclude
that $x \in Y$
except with failure
probability 2^{-40}

$Y = (y_1, \ldots, y_n)$

$BF_Y = (b_1, \ldots, b_N)$

$N = 58\,n$

Google

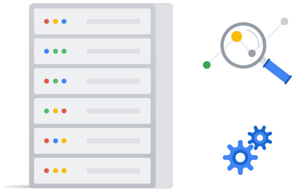
PIR + BF

User

X	V
x_1	v_1
...	...
x_m	v_m

Y	W
y_1	w_1
...	...
...	...
...	...
...	...
y_n	w_n

Server



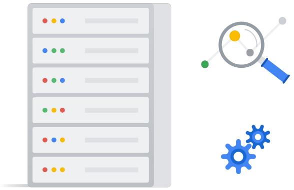
PIR + BF

User

x

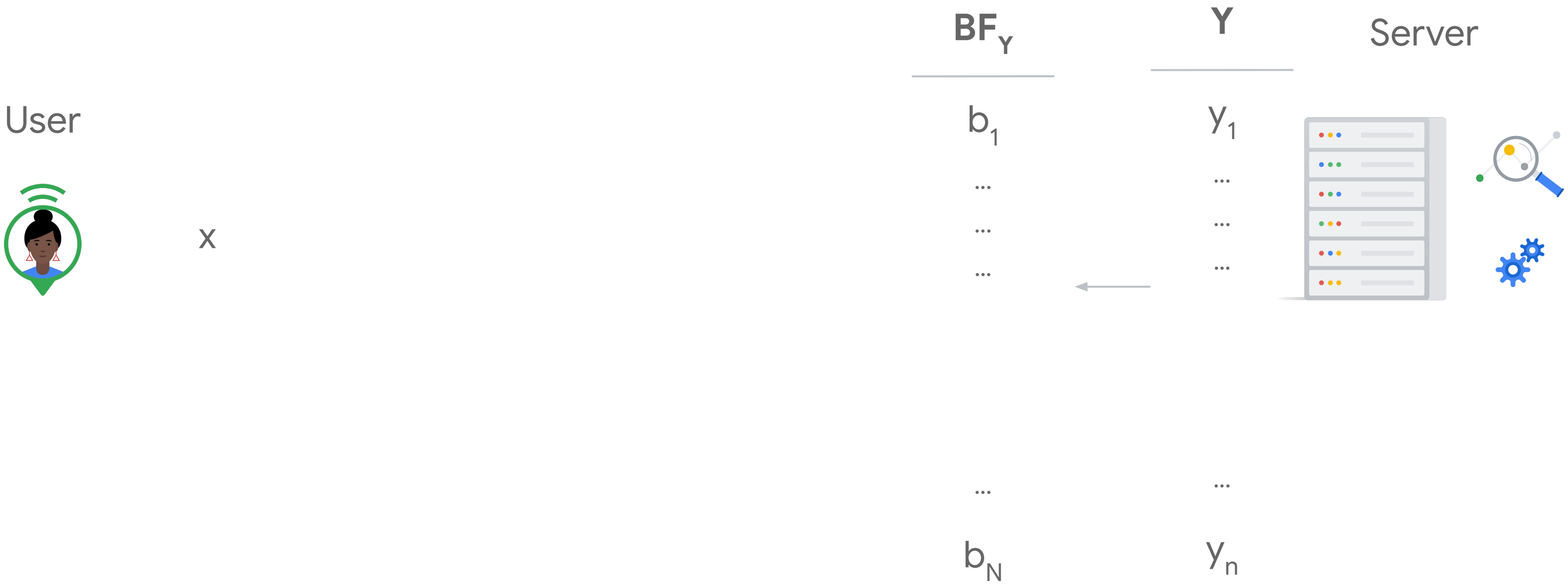
Y
y_1
...
...
...
...
y_n

Server



Google

PIR + BF



PIR + BF

User

x

BF_Y

b_1

…

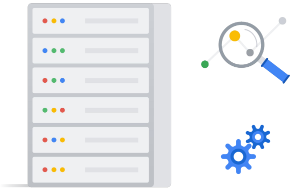
…

…

…

b_N

Server



PIR + BF

User

x

$Enc(h_1(x)), ..., Enc(h_k(x))$ →

BF_Y

b_1

...

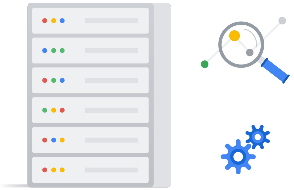
...

...

...

b_N

Server



PIR + BF

User

x

$\text{Enc}(h_1(x)), \ldots, \text{Enc}(h_k(x))$

Process PIR queries

$\text{Enc}(b_{h\_1(x)}), \ldots , \text{Enc}(b_{h\_k(x)})$

BF_Y

b_1

...

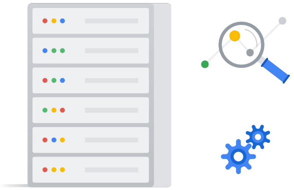
...

...

...

b_N

Server



PIR + BF

User

x

$Enc(h_1(x)), \ldots, Enc(h_k(x))$ →

Process PIR queries

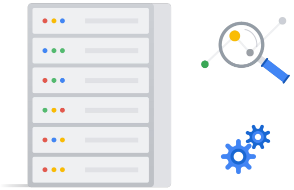
$Enc(b_{h\_1(x)}), \ldots, Enc(b_{h\_k(x)})$

Hom. Sum responses

$Enc(\Sigma\, b_{h\_i(x)})$

BF_Y
b_1
...
...
...
...
b_N

Server



PIR + BF

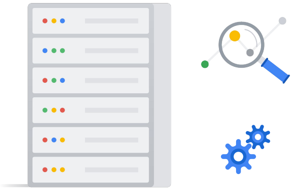
User

x

$Enc(h_1(x)), \ldots, Enc(h_k(x))$

$\mathbf{BF_Y}$

Server



$\mathbf{BF_Y}$
b_1
...
...
...
...
b_N

Process PIR queries

$Enc(b_{h\_1(x)}), \ldots , Enc(b_{h\_k(x)})$

Hom. Sum responses and mask

$Enc(\Sigma\, b_{h\_i(x)} + r_2)$

r_2

PIR + BF

User

x

$\text{Enc}(h_1(x)), \ldots, \text{Enc}(h_k(x))$

BF_Y

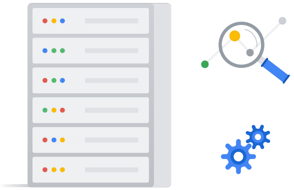
b_1

...

...

...

Server



Process PIR queries

$\text{Enc}(b_{h\_1(x)}), \ldots, \text{Enc}(b_{h\_k(x)})$

Hom. Sum responses
and mask

$\text{Enc}(\Sigma\, b_{h\_i(x)} + r_2)$

...

b_N

r_2

PIR + BF

User

x

$Enc(h_1(x)), ..., Enc(h_k(x))$ →

Process PIR queries

$Enc(b_{h\_1(x)}), ... , Enc(b_{h\_k(x)})$

Hom. Sum responses
and mask

$Enc(\Sigma\, b_{h\_i(x)} + r_2)$ ←

r_1 = Decrypt and subtract k

BF_Y

b_1

...

...

...

...

b_N

Server

r_2

PIR + BF

$r_1 = r_2$ if and only if $x \in Y$

User

x

$\text{Enc}(h_1(x)), \ldots, \text{Enc}(h_k(x))$

Process PIR queries

$\text{Enc}(b_{h\_1(x)}), \ldots, \text{Enc}(b_{h\_k(x)})$

Hom. Sum responses and mask

$\text{Enc}(\Sigma\, b_{h\_i(x)} + r_2)$

r_1 = Decrypt and subtract k

BF_Y

b_1

...

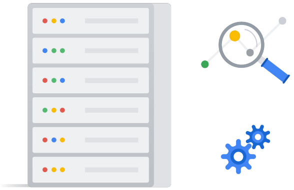
...

...

...

b_N

Server



r_2

Google

PIR + BF

$r_1 = r_2$ if and only if $x \in Y$

(except w.p. 2^{-40})

User

x

$Enc(h_1(x)), \ldots, Enc(h_k(x))$

Process PIR queries

$Enc(b_{h\_1(x)}), \ldots, Enc(b_{h\_k(x)})$

Hom. Sum responses
and mask

$Enc(\Sigma\, b_{h\_i(x)} + r_2)$

r_1 = Decrypt and subtract k

BF_Y

b_1

...

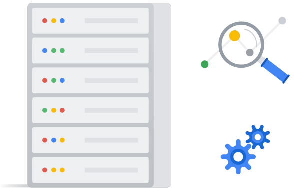
...

...

...

b_N

Server



r_2

Google

PIR + BF

$r_1 = r_2$ if and only if $x \in Y$

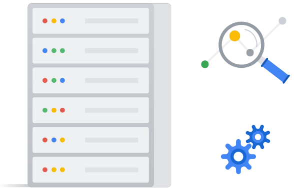
(except w.p. 2^{-40})

r_2

Server

r_1

User



Google

Associated Values?

Google

Associated Values: Garbled Bloom Filter

Google

Garbled Bloom Filter (GBF)

$$Y = (y_1, \dots, y_n)$$

$$W = (w_1, \dots, w_n)$$

$$\downarrow$$

$$GBF_{Y,W} = (g_1, \dots, g_N)$$

Garbled Bloom Filter (GBF)

x

$h_1(x), \ldots, h_k(x)$

$Y = (y_1, \ldots, y_n)$

$W = (w_1, \ldots, w_n)$

$GBF_{Y,W} = (g_1, \ldots, g_N)$

Garbled Bloom Filter (GBF)

x

$h_1(x), \ldots, h_k(x)$

If $x \in Y$ then

$\Sigma_i \, GBF_{Y,W}[h_i(x)] = W[x]$

$Y = (y_1, \ldots, y_n)$

$W = (w_1, \ldots, w_n)$

$GBF_{Y,W} = (g_1, \ldots, g_N)$

Garbled Bloom Filter (GBF)

x

$h_1(x), \ldots, h_k(x)$

If $x \in Y$ then

$\Sigma_i \, GBF_{Y,W}[h_i(x)] = W[x]$

If $x \notin Y$ then

$\Sigma_i \, GBF_{Y,W}[h_i(x)] =$?

$Y = (y_1, \ldots, y_n)$

$W = (w_1, \ldots, w_n)$

$GBF_{Y,W} = (g_1, \ldots, g_N)$

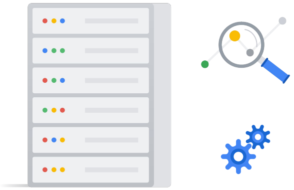
PIR + GBF

User

x v

Y	W
y_1	w_1
...	...
...	...
...	...
...	...
y_n	w_n

Server



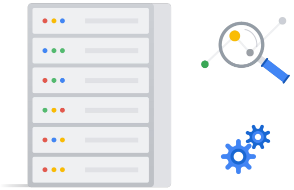
PIR + GBF

User

x v

$GBF_{Y,W}$	Y	W
g_1	y_1	w_1
...	...	...
...	...	...
...	...	...
...	...	...
g_N	y_n	w_n

Server



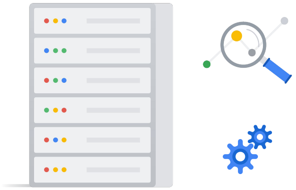
PIR + GBF

User

x v

$GBF_{Y,W}$

Server



g_1

…

…

…

…

g_N

PIR + GBF

User

x v

Enc(h_1(x)), …, Enc(h_k(x))

$GBF_{Y,W}$

g_1

…

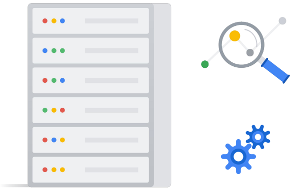
…

…

…

g_N

Server



PIR + GBF

User

x v

$\text{Enc}(h_1(x)), \ldots, \text{Enc}(h_k(x))$

Process PIR queries

$\text{Enc}(g_{h\_1(x)}), \ldots, \text{Enc}(g_{h\_k(x)})$

$\text{GBF}_{Y,W}$

g_1

...

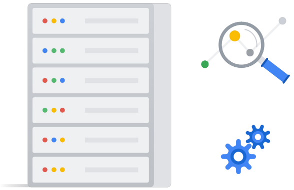
...

...

...

g_N

Server



PIR + GBF

User

x   v

$\mathrm{Enc}(h_1(x)), \ldots, \mathrm{Enc}(h_k(x))$

Process PIR queries

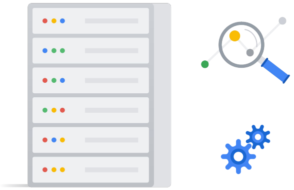
$\mathrm{Enc}(g_{h\_1(x)}), \ldots, \mathrm{Enc}(g_{h\_k(x)})$

Hom. Sum responses

$\mathrm{Enc}(\Sigma\, g_{h\_i(x)})$

$\mathbf{GBF_{Y,W}}$
g_1
…
…
…
…
g_N

Server



PIR + GBF

User

x v

Enc(v)

Enc($h_1(x)$), …, Enc($h_k(x)$)

Process PIR queries

Enc($g_{h\_1(x)}$), … , Enc($g_{h\_k(x)}$)

Hom. Sum responses

Enc($\Sigma\, g_{h\_i(x)}$)

$GBF_{Y,W}$

g_1

…

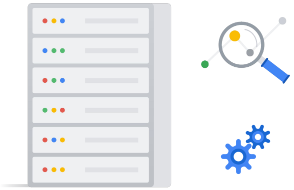
…

…

…

g_N

Server



PIR + GBF

User

x v

Enc(v)

$Enc(h_1(x)), \ldots, Enc(h_k(x))$

$GBF_{Y,W}$

g_1

…

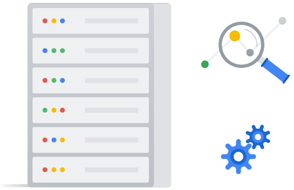
…

…

…

g_N

Server



Process PIR queries

$Enc(g_{h\_1(x)}), \ldots, Enc(g_{h\_k(x)})$

Hom. Sum responses

Multiply v

$Enc(v * \Sigma g_{h\_i(x)})$

PIR + GBF

User

x   v

$Enc(v)$

$Enc(h_1(x)), \ldots, Enc(h_k(x))$

Process PIR queries

$Enc(g_{h\_1(x)}), \ldots, Enc(g_{h\_k(x)})$

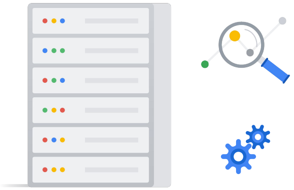
Hom. Sum responses
Multiply v and mask

$Enc(v * \Sigma\, g_{h\_i(x)} - s_2)$

$GBF_{Y,W}$

g_1
...
...
...
...
g_N

Server



s_2

PIR + GBF

User

x v

Enc(v)

$Enc(h_1(x)), \ldots, Enc(h_k(x))$

$GBF_{Y,W}$

Server

g_1

…

…

…

…

g_N

Process PIR queries

$Enc(g_{h\_1(x)}), \ldots, Enc(g_{h\_k(x)})$

Hom. Sum responses

Multiply v and mask

$Enc(v * \Sigma\, g_{h\_i(x)} - s_2)$

s_2

s_1 = Decrypt

PIR + GBF

$s_1 + s_2 = V[x] * W[x]$ if $x \in Y$
$= ?$ otherwise

User

x v

Enc(v)

$Enc(h_1(x)), \ldots, Enc(h_k(x))$

$GBF_{Y,W}$

g_1

...

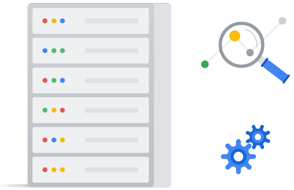
...

...

...

g_N

Server



Process PIR queries

$Enc(g_{h\_1(x)}), \ldots , Enc(g_{h\_k(x)})$

Hom. Sum responses
Multiply v and mask
$Enc(v * \Sigma\, g_{h\_i(x)} - s_2)$

s_2

s_1 = Decrypt

Google

Putting it together: PIR with Default

$r_1 = r_2$ if and only if $x \in Y$

$s_1 + s_2 = V[x] * W[x]$ if $x \in Y$
$= \ ?$ otherwise

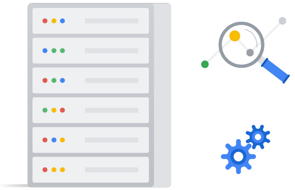
User

x v

r_1 s_1

Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

Server



r_2 s_2

Putting it together: PIR with Default

$r_1 = r_2$ if and only if $x \in Y$

$s_1 + s_2 = V[x] * W[x]$ if $x \in Y$
$= \ ?$ otherwise

User

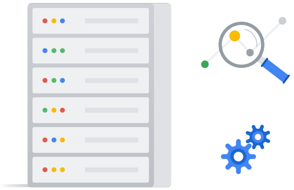
x v

r_1 s_1

"Generic" MPC protocol

Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

Server



r_2 s_2

t_1

t_2

Google

Putting it together: PIR with Default

$r_1 = r_2$ if and only if $x \in Y$

$s_1 + s_2 = V[x] * W[x]$ if $x \in Y$
$= ?$ otherwise

User

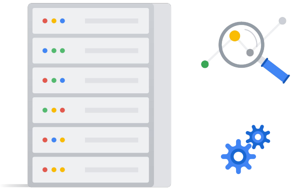
x v

r_1 s_1

"Generic" MPC protocol

Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

Server



r_2 s_2

t_1

$t_1 + t_2 = V[x] * W[x]$ if $x \in Y$
$= 0$ otherwise

t_2

Putting it together: PIR with Default

$r_1 = r_2$ if and only if $x \in Y$

$s_1 + s_2 = V[x] * W[x]$ if $x \in Y$
$= \ ?$ otherwise

User

x v

r_1 s_1

"Generic" MPC protocol

default

Server

Y	W
y_1	w_1
…	…
…	…
…	…
…	…
y_n	w_n

r_2 s_2

t_1

$t_1 + t_2 = V[x] * W[x]$ if $x \in Y$
$=$ default otherwise

t_2

Optimizations

Optimizations

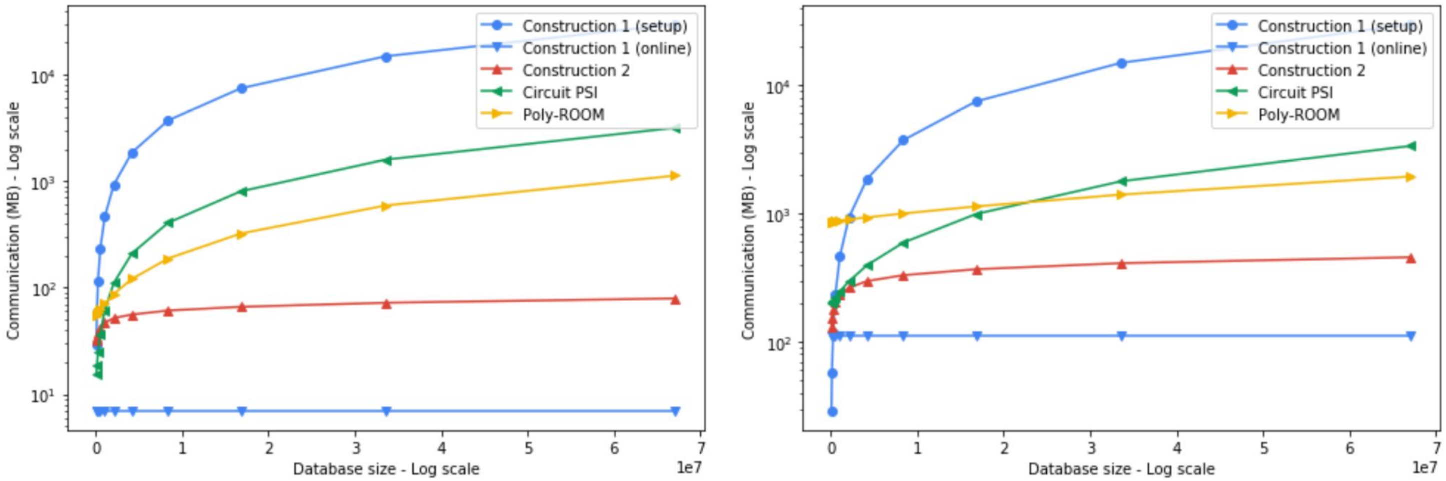
- Slotting/ Batching:
    - Enables multiple PIR queries to be executed in parallel.

Optimizations

- Slotting/ Batching:
  - Enables multiple PIR queries to be executed in parallel.
- Cuckoo hashing inputs:
  - Standard technique to group the inputs into smaller buckets and execute the protocol only over each bucket.
  - Huge computational savings on the server at a minimal increase in client costs.

Experimental Costs

Communication Costs



(a) $t = 2^8$ (b) $t = 2^{12}$

Figure 7: Communication cost of t PIR-with-Default queries, for increasing database sizes n and fixed number t.

The presented construction is the red line. “t” is the number of client queries. Communication costs grow more slowly as the Server’s database increases.

Cost Table

Parameters		Construction 1				Construction 2		Circuit PSI [PSTY19]		Poly-ROOM [SGRP19]		PJC+RLWE [IKN+20]	
		Setup		Online		Online		Online		Online		Online	
n	t	Comm. (MB)	Time (/query)	Comm. (MB)	Time (/query)	Comm. (MB)	Time (/query)	Comm. (MB)	Time (/query)	Comm. (MB)	Time (/query)	Comm. (MB)	Time (/query)
2^{16}	2^8	29	35ms	7	2.43ms	27	673ms	5	11.79ms	55	59ms*	$3^{\dagger}$	$44.8ms^{\dagger}$
	2^{12}	29	2.19ms	112	1.03ms	120	34ms	30	0.93ms	863	3.5ms*	$3^{\dagger}$	$2.97ms^{\dagger}$
	2^{16}	29	0.14ms	1794	0.72ms	801	2ms	472	0.13ms	13788	2.2ms*	$6^{\dagger}$	$0.36ms^{\dagger}$
2^{20}	2^8	465	539ms	7	2.43ms	29	11821ms	51	178ms	71	–	$40^{\dagger}$	$713ms^{\dagger}$
	2^{12}	465	34ms	112	1.03ms	213	521ms	76	11.31ms	878	–	$40^{\dagger}$	$44.7ms^{\dagger}$
	2^{16}	465	2.11ms	1794	0.72ms	1821	34ms	522	0.78ms	13837	–	$44^{\dagger}$	$2.97ms^{\dagger}$
2^{25}	2^8	14885	17252ms	7	2.43ms	44	370s	1582	5668ms	591	–	$1272^{\dagger}$	$22838ms^{\dagger}$
	2^{12}	14885	1078ms	112	1.03ms	379	15.8s	1607	354ms	1401	–	$1272^{\dagger}$	$1427ms^{\dagger}$
	2^{16}	14885	67ms	1794	0.72ms	3704	1.1s	2180	22.22ms	14391	–	$1276^{\dagger}$	$89ms^{\dagger}$

Machine: single core of Intel(R) Xeon(R) CPU E5-2696 v3 @ 2.30GHz. For all constructions and $n = 2^{25}$, times have been estimated from microbenchmarks of the core operations, and fixed cost for a random access was assumed.

* The times for Poly-ROOM are taken from [SGRP19, Fig. 17], initially provided for a database $n = 50,000$ and a number of queries $t = 5,000$ and $50,000$. Unknown machine.

† Although PJC+RLWE does not achieve the PIR-with-Default functionality, we report it for comparison purpose. Timings are estimated from microbenchmarks of NIST-P256, and RLWE-encryption with degree 2048 and 62 bit modulus.

Table 2: Communication and computation costs of PIR-with-Default with elements of 32 bits. Running time is amortized over the number of client queries.

Cost Table

Parameters		Construction 1				Construction 2		Circuit PSI [PSTY19]		Poly-ROOM [SGRP19]		PJC+RLWE [IKN+20]	
		Setup		Online		Online		Online		Online		Online	
n	t	Comm. (MB)	Time (/query)	Comm. (MB)	Time (/query)	Comm. (MB)	Time (/query)	Comm. (MB)	Time (/query)	Comm. (MB)	Time (/query)	Comm. (MB)	Time (/query)
2^{16}	2^8	29	35ms	7	2.43ms	27	673ms	5	11.79ms	55	59ms*	$3^\dagger$	$44.8\text{ms}^\dagger$
	2^{12}	29	2.19ms	112	1.03ms	120	34ms	30	0.93ms	863	3.5ms*	$3^\dagger$	$2.97\text{ms}^\dagger$
	2^{16}	29	0.14ms	1794	0.72ms	801	2ms	472	0.13ms	13788	2.2ms*	$6^\dagger$	$0.36\text{ms}^\dagger$
2^{20}	2^8	465	539ms	7	2.43ms	29	11821ms	51	178ms	71	–	$40^\dagger$	$713\text{ms}^\dagger$
	2^{12}	465	34ms	112	1.03ms	213	521ms	76	11.31ms	878	–	$40^\dagger$	$44.7\text{ms}^\dagger$
	2^{16}	465	2.11ms	1794	0.72ms	1821	34ms	522	0.78ms	13837	–	$44^\dagger$	$2.97\text{ms}^\dagger$
2^{25}	2^8	14885	17252ms	7	2.43ms	44	370s	1582	5668ms	591	–	$1272^\dagger$	$22838\text{ms}^\dagger$
	2^{12}	14885	1078ms	112	1.03ms	379	15.8s	1607	354ms	1401	–	$1272^\dagger$	$1427\text{ms}^\dagger$
	2^{16}	14885	67ms	1794	0.72ms	3704	1.1s	2180	22.22ms	14391	–	$1276^\dagger$	$89\text{ms}^\dagger$

Machine: single core of Intel(R) Xeon(R) CPU E5-2696 v3 @ 2.30GHz. For all constructions and $n = 2^{25}$, times have been estimated from microbenchmarks of the core operations, and fixed cost for a random access was assumed.

* The times for Poly-ROOM are taken from [SGRP19, Fig. 17], initially provided for a database $n = 50,000$ and a number of queries $t = 5,000$ and $50,000$. Unknown machine.

† Although PJC+RLWE does not achieve the PIR-with-Default functionality, we report it for comparison purpose. Timings are estimated from microbenchmarks of NIST-P256, and RLWE-encryption with degree 2048 and 62 bit modulus.

Table 2: Communication and computation costs of PIR-with-Default with elements of 32 bits. Running time is amortized over the number of client queries.

Monetary Costs

Parameters		Construction 1		Construction 2		Circuit PSI		PJC+RLWE	
n	t	Client	Server	Client	Server	Client	Server	Client	Server
2^{16}	2^8	0.14	0.11	0.11	0.15	0.06	0.06	0.01	0.01
	2^{12}	0.55	0.11	0.47	0.51	0.78	0.78	0.01	0.01
	2^{16}	7.14	0.11	3.13	3.17	12.51	12.51	0.03	0.03
2^{20}	2^8	1.84	1.84	0.11	0.95	0.24	0.25	0.18	0.18
	2^{12}	2.26	1.84	0.83	1.42	0.97	0.98	0.18	0.18
	2^{16}	8.84	1.84	7.11	7.73	12.7	12.72	0.2	0.2
2^{25}	2^8	58.17	58.76	0.17	26.48	6.22	6.62	5.78	5.78
	2^{12}	58.58	58.76	1.48	19.46	6.94	7.34	5.78	5.78
	2^{16}	65.17	58.76	14.47	34.49	19.18	19.58	5.8	5.8

Table 3: Total monetary cost in USD cents of PIR-with-Default with elements of 32 bits, using GCP pricing for network and compute costs (see Table 5). Costs are totals across t queries including network cost (divided equally amongst client and server), and computation costs for both client and server including setup.

Monetary Costs

Parameters		Construction 1		Construction 2		Circuit PSI		PJC+RLWE	
n	t	Client	Server	Client	Server	Client	Server	Client	Server
2^{16}	2^8	0.14	0.11	0.11	0.15	0.06	0.06	0.01	0.01
	2^{12}	0.55	0.11	0.47	0.51	0.78	0.78	0.01	0.01
	2^{16}	7.14	0.11	3.13	3.17	12.51	12.51	0.03	0.03
2^{20}	2^8	1.84	1.84	0.11	0.95	0.24	0.25	0.18	0.18
	2^{12}	2.26	1.84	0.83	1.42	0.97	0.98	0.18	0.18
	2^{16}	8.84	1.84	7.11	7.73	12.7	12.72	0.2	0.2
2^{25}	2^8	58.17	58.76	0.17	26.48	6.22	6.62	5.78	5.78
	2^{12}	58.58	58.76	1.48	19.46	6.94	7.34	5.78	5.78
	2^{16}	65.17	58.76	14.47	34.49	19.18	19.58	5.8	5.8

Table 3: Total monetary cost in USD cents of PIR-with-Default with elements of 32 bits, using GCP pricing for network and compute costs (see Table 5). Costs are totals across t queries including network cost (divided equally amongst client and server), and computation costs for both client and server including setup.

Extensions

Other functionalities

$$\sum_{x \in X \cap Y} V[x] \cdot W[x] + \varepsilon$$

Other functionalities

$$\sum_{x \in X \cap Y} V[x] \cdot W[x] \; + \varepsilon$$

$$\sum_{x \in X \cap Y} f(V[x], W[x]) \; + \varepsilon$$

For f supported by Homomorphic Encryption

Other functionalities

$$\sum_{x \in X \cap Y} V[x] \cdot W[x] + \varepsilon$$

$$\sum_{x \in X \cap Y} f(V[x],W[x]) + \varepsilon$$

For f supported by Homomorphic Encryption

$$G(\{ f(V[x],W[x]) \}_{x \in X \cap Y}) + \varepsilon$$

For G supported by the secret sharing scheme
(Or, with more cost, any generic G)

More recent works

More recent works

- Vector-OLE based PSI
  - May be an improvement over Circuit PSI for inner-join PJC
- Labeled PSI from Fully Homomorphic Encryption with Malicious Security
  - Builds on Chen et al "Labeled PSI from Fully Homomorphic Encryption with Malicious Security"
  - Targets the asymmetric setting, with label retrieval.

Thank You!

Google