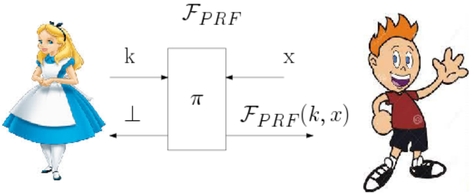
Batched Related Key Oblivious Psuedo Random Function

Abhishek Kumar
Supervised by: Dr. Bhavana Kanukurthi

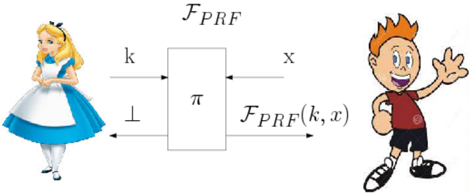
July 18, 2017

Table of contents

Introduction

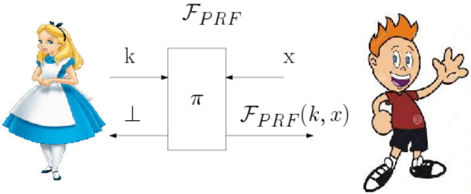


Introduction



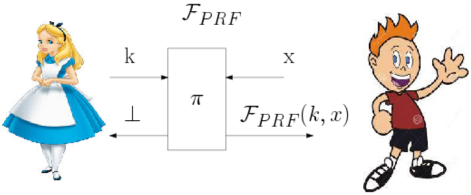
- In this talk we will talk about a protocol for evaluating a large batch OPRF instances.

Introduction



- In this talk we will talk about a protocol for evaluating a large batch OPRF instances.
- We will consider semi-honest settings.

Introduction



- In this talk we will talk about a protocol for evaluating a large batch OPRF instances.
- We will consider semi-honest settings.
- The above protocol is due to [KKRT16] called Batched Related Key OPRF(BaRK-OPRF).

Motivation

- This work significantly improves the state-of-the-art Private Set Intersection(PSI) protocol of [PSSZ15].

Motivation

- This work significantly improves the state-of-the-art Private Set Intersection(PSI) protocol of [PSSZ15].
- Among the problems of secure computation, PSI is most strongly motivated by practice.☺

Motivation

- This work significantly improves the state-of-the-art Private Set Intersection(PSI) protocol of [PSSZ15].
- Among the problems of secure computation, PSI is most strongly motivated by practice.☺
- It is used for measuring Ad Conversion rate, Security incident information sharing, private contact discovery.

Motivation

- This work significantly improves the state-of-the-art Private Set Intersection(PSI) protocol of [PSSZ15].
- Among the problems of secure computation, PSI is most strongly motivated by practice.☺
- It is used for measuring Ad Conversion rate, Security incident information sharing, private contact discovery.

Related Work

- OPRFs were introduced by [FIPR05] but their protocol uses expensive public key operations. Also they use number of OTs proportional to bit length of PRF input. Thus not useful if we have to carry out large number of OPRF instances.☹

Related Work

- OPRFs were introduced by [FIPR05] but their protocol uses expensive public key operations. Also they use number of OTs proportional to bit length of PRF input. Thus not useful if we have to carry out large number of OPRF instances.☹
- The protocol of [CNS07] constructs an OPRF from unique blind signature schemes.

Notations

- We denote vectors in bold for eg. $\mathbf{a}$.
- OT_l^m means m instances of OTs of l-bit strings.
- We denote Matrices in capital say M.
- For matrix M
  - M_i means i^{th} row of M
  - M^j means j^{th} coloumn of M.
- For vector $\mathbf{a}$ and bit b, $\mathbf{a} \cdot b$ means $\mathbf{a}$ if $b = 1$ else $\mathbf{0}$.
- For vectors $\mathbf{a}$ and bit $\mathbf{b}$, $\mathbf{a} \cdot \mathbf{b}$ means bitwise AND.

Correlation Robustness

Definition (1)

*Let H be a hash function. Then H is a d-**Hamming correlation robust** if for any strings* $\mathbf{z_1} \ldots . \mathbf{z_m} \in \{0,1\}^*$ *and* $\mathbf{a_1}, \ldots . \mathbf{a_m}, \mathbf{b_1}, \ldots . \mathbf{b_m} \in \{0,1\}^n$ *with* $\|\mathbf{b_i}\|_H \geq d$*, the following distribution, induced by random sampling of* $\mathbf{s} \longleftarrow \{0,1\}^n$*, is psuedo-random :*

$$H(\mathbf{z_1}||\mathbf{a_1} \oplus [\mathbf{b_1} \cdot \mathbf{s}]), \ldots .., H(\mathbf{z_m}||\mathbf{a_m} \oplus [\mathbf{b_m} \cdot \mathbf{s}])$$

"." denotes bitwise-AND

Psuedo-Random Codes

Definition (2)

Let $\mathcal{C}$ be a family of functions. We say that $\mathcal{C}$ is a (d, ϵ) **psuedorandom code(PRC)** *if for all strings $x \neq x'$,*

$$\Pr_{C \leftarrow \mathcal{C}} [\|C(x) \oplus C(x')\|_H < d] \leq 2^{-\epsilon}$$

Psuedo-Random Codes

Definition (2)

Let $\mathcal{C}$ be a family of functions. We say that $\mathcal{C}$ is a (d, ϵ) **psuedorandom code(PRC)** *if for all strings $x \neq x'$,*

$$\Pr_{C \leftarrow \mathcal{C}} [\|C(x) \oplus C(x')\|_H < d] \leq 2^{-\epsilon}$$

Lemma (1)

Suppose $G : \{0,1\}^\kappa \times \{0,1\}^* \to \{0,1\}^n$ is a psuedorandom function. Define $\mathcal{C} = \{G(s, \cdot) \mid s \in \{0,1\}^\kappa\}$. Then $\mathcal{C}$ is a (d, ϵ)-psuedorandom code where :

$$2^{-\epsilon} = 2^{-n} \sum_{i=0}^{d-1} \binom{n}{i} + \vartheta(\kappa)$$

High Level Idea

- We see 1-out-of-∞ OT as OPRF. The intuition begind this is the fact that Sender has ability to evaluate the function at any point but remains oblivious to Receiver's choice. Also Receiver can't evaluate function at any other point as it doesn't have access to key.

High Level Idea

- We see 1-out-of-∞ OT as OPRF. The intuition begind this is the fact that Sender has ability to evaluate the function at any point but remains oblivious to Receiver's choice. Also Receiver can't evaluate function at any other point as it doesn't have access to key.
- However in the construction as we shall see, more information is leaked than required. Hence we will meet the definition of relaxed OPRF as defined in [FIPR05].

relaxed PRF

Definition (3)

F is said to be a relaxed PRF if there is another $\widetilde{F}$, such that $F(key,r)$ can be efficiently computed given just $\widetilde{F}(key, r)$.

relaxed PRF

Definition (3)

F is said to be a relaxed PRF if there is another $\widetilde{F}$*, such that F(key,r) can be efficiently computed given just* $\widetilde{F}(key, r)$*.*

Definition (4)

Let F be a relaxed PRF with output length v, for which we can write the key $= (k^*, k)$*. Then F has* $\mathbf{m-related-key-PRF(m-RK-PRF)security}$ *if the advantage of any PPT adversary in the following game is negligible :*

Security Game

1. The adversary chooses a large no. of strings $x_1, ...x_n$ which it will query to $\widetilde{F}$. $x_1, ...x_n \in \{0, 1\}^*$.

Security Game

1. The adversary chooses a large no. of strings $x_1, ...x_n$ which it will query to $\widetilde{F}$. $x_1, ...x_n \in \{0,1\}^*$.
2. Then it selects m pairs $(p_1, y_1), ...(p_m, y_m)$, where $p_i \neq x_{j_i}$, $y_1, ...y_m \in \{0,1\}^*$

Security Game

1. The adversary chooses a large no. of strings $x_1, ...x_n$ which it will query to $\widetilde{F}$. $x_1, ...x_n \in \{0,1\}^*$.
2. Then it selects m pairs $(p_1, y_1), ...(p_m, y_m)$, where $p_i \neq x_{j_i}$, $y_1, ...y_m \in \{0,1\}^*$
3. Challenger chooses $k^*, k_1, k_2, ...k_n$ such that each pair (k^*, k_i) can be used as key to F.

Security Game

1. The adversary chooses a large no. of strings $x_1, ...x_n$ which it will query to $\widetilde{F}$. $x_1, ...x_n \in \{0,1\}^*$.
2. Then it selects m pairs $(p_1, y_1), ...(p_m, y_m)$, where $p_i \neq x_{j_i}$, $y_1, ...y_m \in \{0,1\}^*$
3. Challenger chooses $k^*, k_1, k_2, ...k_n$ such that each pair (k^*, k_i) can be used as key to F.

Remark

Pair (K^*, K_i) will be used as *key* to F for input x_i for $i \in [n]$. We assume that $p_1, ...p_m \in [n]$. This is because $p_i's$ are basically meant to determine the key (k^*, k_{p_i}) that will be used when input to F is $y_i's$ for $y \in [m]$.

Remark

$y_i \neq x_{p_i}$ for $i \in [m]$. If this is not true then adversary will win trivially, as it means that adversary is choosing is a string which has been queried before with has been queried before with same key hence it can easily distinguish from random.

Game continued...

4. Challenger tosses a coin $b \leftarrow \{0, 1\}$.

Game continued...

4. Challenger tosses a coin $b \leftarrow \{0, 1\}$.
   (a) If $b = 0$, challenger outputs $\{\widetilde{F}((k^*, k_i), x_i)\}_i$ for $i \in [n]$ and $\{F((k^*, k_{p_l}), y_l)\}_l$ for $l \in [m]$.

Game continued...

4. Challenger tosses a coin $b \leftarrow \{0,1\}$.
   (a) If $b = 0$, challenger outputs $\{\widetilde{F}((k^*, k_i), x_i)\}_i$ for $i \in [n]$ and $\{F((k^*, k_{p_l}), y_l)\}_l$ for $l \in [m]$.
   (b) If $b = 1$, challenger outputs $\{\widetilde{F}((k^*, k_i), x_i)\}_i$ for $i \in [n]$ and $\{z_l\}_l$ for $l \in [m]$ where $z_l \in_{randomly} \{0,1\}^v$.

Game continued...

4. Challenger tosses a coin $b \leftarrow \{0, 1\}$.
   (a) If $b = 0$, challenger outputs $\{\widetilde{F}((k^*, k_i), x_i)\}_i$ for $i \in [n]$ and $\{F((k^*, k_{p_l}), y_l)\}_l$ for $l \in [m]$.
   (b) If $b = 1$, challenger outputs $\{\widetilde{F}((k^*, k_i), x_i)\}_i$ for $i \in [n]$ and $\{z_l\}_l$ for $l \in [m]$ where $z_l \in_{randomly} \{0, 1\}^v$.
5. Adversary outputs a bit b'. Adversary wins if $b' = b$

Game continued...

4. Challenger tosses a coin $b \leftarrow \{0,1\}$.
   (a) If $b = 0$, challenger outputs $\{\widetilde{F}((k^*, k_i), x_i)\}_i$ for $i \in [n]$ and $\{F((k^*, k_{p_l}), y_l)\}_l$ for $l \in [m]$.
   (b) If $b = 1$, challenger outputs $\{\widetilde{F}((k^*, k_i), x_i)\}_i$ for $i \in [n]$ and $\{z_l\}_l$ for $l \in [m]$ where $z_l \in_{randomly} \{0,1\}^v$.
5. Adversary outputs a bit b'. Adversary wins if $b' = b$

The PRF is secure if $\Pr[b = b'] \leq \frac{1}{2} +$ *negligible*

Game continued...

4. Challenger tosses a coin $b \leftarrow \{0,1\}$.
   (a) If $b = 0$, challenger outputs $\{\widetilde{F}((k^*, k_i), x_i)\}_i$ for $i \in [n]$ and $\{F((k^*, k_{p_l}), y_l)\}_l$ for $l \in [m]$.
   (b) If $b = 1$, challenger outputs $\{\widetilde{F}((k^*, k_i), x_i)\}_i$ for $i \in [n]$ and $\{z_l\}_l$ for $l \in [m]$ where $z_l \in_{randomly} \{0,1\}^v$.
5. Adversary outputs a bit b'. Adversary wins if $b' = b$

The PRF is secure if $\Pr[b = b'] \leq \frac{1}{2} +$ *negligible*

Remark

Intuitively the *PRF* is instantiated with n related keys(sharing k^* value). The adversary learns the relaxed output on one chosen input for each key. Then any m additional *PRF* outputs are indistinguishable from random.

- Let **s** denote a random vector chosen by sender.

- Let $\mathbf{s}$ denote a random vector chosen by sender.
- $\mathbf{q}_j$ is a vector known to sender. j is used for indexing and known to both.

- Let $\mathbf{s}$ denote a random vector chosen by sender.
- $\mathbf{q}_j$ is a vector known to sender. j is used for indexing and known to both.
- r is the evaluation point known to receiver only.

- Let $\mathbf{s}$ denote a random vector chosen by sender.
- $\mathbf{q}_j$ is a vector known to sender. j is used for indexing and known to both.
- r is the evaluation point known to receiver only.

Lemma (2)

Let $\mathcal{C}$ be a $(d, \epsilon + \log m) - PRC$ let H be $d -$ hamming correlation robust hash function. Let us define following relaxed PRF for $C \leftarrow_{randomly} \mathcal{C}$:

$$F\Big(((C,s),(q_j,j)),r\Big) = H(j||q_j \oplus [C(r)\cdot s])$$
$$\widetilde{F}\Big(((C,s),(q_j,j)),r\Big) = (j, C, q_j \oplus [C(r)\cdot s])$$

Then F has $\mathbf{m-RK-PRF}$ security .

Ideal Functionality we want to achieve :

- There are two parties, psuedorandom function F and corresponding $\widetilde{F}$ a number m of instances.

Ideal Functionality we want to achieve :

- There are two parties, psuedorandom function F and corresponding $\widetilde{F}$ a number m of instances.
- On input $(r_1, r_2, \ldots\ldots r_m)$ from receiver :
  - Choose random components for seeds to PRF : $k^*, k_1, k_2, \ldots\ldots k_m$ and give it to sender.

Ideal Functionality we want to achieve :

- There are two parties, psuedorandom function F and corresponding $\widetilde{F}$ a number m of instances.
- On input $(r_1, r_2, ...r_m)$ from receiver :
  - Choose random components for seeds to PRF : $k^*, k_1, k_2, ...k_m$ and give it to sender.
  - Give $\widetilde{F}((k^*, k_1,), r_1), ...\widetilde{F}((k^*, k_m), r_m)$ to receiver.

BaRK OPRF Protocol

Input of R : m selection strings $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m), r_i \in \{0,1\}^*, i \in [m]$.

BaRK OPRF Protocol

Input of R : m selection strings $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m), r_i \in \{0,1\}^*, i \in [m]$.
Parameters :

- A (κ, ϵ)-PRC family $\mathcal{C}$ with output length $k = g(\kappa)$.

BaRK OPRF Protocol

Input of R : m selection strings $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m), r_i \in \{0,1\}^*, i \in [m]$.
Parameters :

- A (κ, ϵ)-PRC family $\mathcal{C}$ with output length $k = g(\kappa)$.
- A $\kappa-$Hamming Correlation Robust $H : [m] \times \{0,1\}^k \to \{0,1\}^v$

BaRK OPRF Protocol

Input of R : m selection strings $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m), r_i \in \{0,1\}^*, i \in [m]$.
Parameters :

- A (κ, ϵ)-PRC family $\mathcal{C}$ with output length $k = g(\kappa)$.
- A $\kappa-$Hamming Correlation Robust $H : [m] \times \{0,1\}^k \rightarrow \{0,1\}^v$
- An Ideal OT_m^k primitive.

BaRK OPRF Protocol

Input of R : m selection strings $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m), r_i \in \{0,1\}^*, i \in [m]$.
Parameters :

- A (κ, ϵ)-PRC family $\mathcal{C}$ with output length $k = g(\kappa)$.
- A $\kappa-$Hamming Correlation Robust $H : [m] \times \{0,1\}^k \to \{0,1\}^v$
- An Ideal OT_m^k primitive.

Protocol :

BaRK OPRF Protocol

Input of R : m selection strings $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m), r_i \in \{0,1\}^*, i \in [m]$.
Parameters :

- A (κ, ϵ)-PRC family $\mathcal{C}$ with output length $k = g(\kappa)$.
- A $\kappa-$Hamming Correlation Robust $H : [m] \times \{0,1\}^k \rightarrow \{0,1\}^v$
- An Ideal OT_m^k primitive.

Protocol :

1. S chooses a random $C \leftarrow \mathcal{C}$ sends to R.

BaRK OPRF Protocol

Input of R : m selection strings $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m), r_i \in \{0,1\}^*, i \in [m]$.
Parameters :

- A (κ, ϵ)-PRC family $\mathcal{C}$ with output length $k = g(\kappa)$.
- A $\kappa-$Hamming Correlation Robust $H : [m] \times \{0,1\}^k \to \{0,1\}^v$
- An Ideal OT_m^k primitive.

Protocol :

1. S chooses a random $C \leftarrow \mathcal{C}$ sends to R.
2. S chooses $\mathbf{s} \leftarrow \{0,1\}^k$ at random. Let s_i denote i^th bit of $\mathbf{s}$.

BaRK OPRF Protocol

Input of R : m selection strings $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m), r_i \in \{0,1\}^*, i \in [m]$.
Parameters :

- A (κ, ϵ)-PRC family $\mathcal{C}$ with output length $k = g(\kappa)$.
- A $\kappa-$Hamming Correlation Robust $H : [m] \times \{0,1\}^k \to \{0,1\}^v$
- An Ideal OT_m^k primitive.

Protocol :

1. S chooses a random $C \leftarrow \mathcal{C}$ sends to R.
2. S chooses $\mathbf{s} \leftarrow \{0,1\}^k$ at random. Let s_i denote $i^t h$ bit of $\mathbf{s}$.
3. R forms $m \times k$ matrix T_0, T_1 in following way :

BaRK OPRF Protocol

- For $j \in [m]$, choose $\mathbf{T}_{0,j} \leftarrow \{0,1\}^k$ randomly.

BaRK OPRF Protocol

- For $j \in [m]$, choose $\mathbf{T}_{0,j} \leftarrow \{0,1\}^k$ randomly.
- Set $\mathbf{T}_{1,j} = \mathbf{T}_{0,j} \oplus C(\mathbf{r}_j)$

BaRK OPRF Protocol

- For $j \in [m]$, choose $\mathbf{T}_{0,j} \leftarrow \{0,1\}^k$ randomly.
- Set $\mathbf{T}_{1,j} = \mathbf{T}_{0,j} \oplus C(\mathbf{r}_j)$

4. S and R interact with OT_m^k in the following way :

BaRK OPRF Protocol

- For $j \in [m]$, choose $\mathbf{T}_{0,j} \leftarrow \{0,1\}^k$ randomly.
- Set $\mathbf{T}_{1,j} = \mathbf{T}_{0,j} \oplus C(\mathbf{r}_j)$

4. S and R interact with OT_m^k in the following way :
   - S acts as receiver with input $\{s_i\}, i \in [k]$

BaRK OPRF Protocol

- For $j \in [m]$, choose $\mathbf{T}_{0,j} \leftarrow \{0,1\}^k$ randomly.
- Set $\mathbf{T}_{1,j} = \mathbf{T}_{0,j} \oplus C(\mathbf{r}_j)$

4. S and R interact with OT_m^k in the following way :
   - S acts as receiver with input $\{s_i\}, i \in [k]$
   - R acts as sender with input $\{\mathbf{T}_0^i, \mathbf{T}_1^i\}, i \in [k]$

BaRK OPRF Protocol

- For $j \in [m]$, choose $\mathbf{T}_{0,j} \leftarrow \{0,1\}^k$ randomly.
- Set $\mathbf{T}_{1,j} = \mathbf{T}_{0,j} \oplus C(\mathbf{r}_j)$

4. S and R interact with OT_m^k in the following way :
    - S acts as receiver with input $\{s_i\}, i \in [k]$
    - R acts as sender with input $\{\mathbf{T}_0^i, \mathbf{T}_1^i\}, i \in [k]$
    - From the output S receives, S forms $m \times k$ matrix Q such that

$$\mathbf{Q}^i = \mathbf{T}^i_{s_i}, \ \ \textit{for } i \in [k]$$
$$\mathbf{Q}_j = \mathbf{T}_{0,j} \oplus (C(\mathbf{r}_j) \cdot \mathbf{s}) \ \textit{for } j \in [m]$$

BaRK OPRF Protocol

- For $j \in [m]$, choose $\mathbf{T}_{0,j} \leftarrow \{0,1\}^k$ randomly.
- Set $\mathbf{T}_{1,j} = \mathbf{T}_{0,j} \oplus C(\mathbf{r}_j)$

4. S and R interact with OT_m^k in the following way :
   - S acts as receiver with input $\{s_i\}, i \in [k]$
   - R acts as sender with input $\{\mathbf{T}_0^i, \mathbf{T}_1^i\}, i \in [k]$
   - From the output S receives, S forms $m \times k$ matrix Q such that

$$\mathbf{Q}^i = \mathbf{T}_{s_i}^i, \ \ \textit{for } i \in [k]$$
$$\mathbf{Q}_j = \mathbf{T}_{0,j} \oplus (C(\mathbf{r}_j) \cdot \mathbf{s}) \ \textit{for } j \in [m]$$

5. For $j \in [m]$ S outputs *PRF* seed $((C, \mathbf{s}), (j, \mathbf{Q}_j))$.

BaRK OPRF Protocol

- For $j \in [m]$, choose $\mathbf{T}_{0,j} \leftarrow \{0,1\}^k$ randomly.
- Set $\mathbf{T}_{1,j} = \mathbf{T}_{0,j} \oplus C(\mathbf{r}_j)$

4. S and R interact with OT_m^k in the following way :
   - S acts as receiver with input $\{s_i\}, i \in [k]$
   - R acts as sender with input $\{\mathbf{T}_0^i, \mathbf{T}_1^i\}, i \in [k]$
   - From the output S receives, S forms $m \times k$ matrix Q such that

$$\mathbf{Q}^i = \mathbf{T}^i_{s_i}, \ \textit{for } i \in [k]$$
$$\mathbf{Q}_j = \mathbf{T}_{0,j} \oplus (C(\mathbf{r}_j) \cdot \mathbf{s}) \ \textit{for } j \in [m]$$

5. For $j \in [m]$ S outputs *PRF* seed $((C, \mathbf{s}), (j, \mathbf{Q}_j))$.
6. For $j \in [m]$ R outputs relaxed-*PRF* output $(C, j, \mathbf{T}_{0,j})$.

BaRK OPRF Protocol

- For $j \in [m]$, choose $\mathbf{T}_{0,j} \leftarrow \{0,1\}^k$ randomly.
- Set $\mathbf{T}_{1,j} = \mathbf{T}_{0,j} \oplus C(\mathbf{r}_j)$

4. S and R interact with OT_m^k in the following way :
   - S acts as receiver with input $\{s_i\}, i \in [k]$
   - R acts as sender with input $\{\mathbf{T}_0^i, \mathbf{T}_1^i\}, i \in [k]$
   - From the output S receives, S forms $m \times k$ matrix Q such that

$$\mathbf{Q}^i = \mathbf{T}^i_{s_i}, \ \ for \ i \in [k]$$
$$\mathbf{Q}_j = \mathbf{T}_{0,j} \oplus (C(\mathbf{r}_j) \cdot \mathbf{s}) \ for \ j \in [m]$$

5. For $j \in [m]$ S outputs *PRF* seed $((C, \mathbf{s}), (j, \mathbf{Q}_j))$.
6. For $j \in [m]$ R outputs relaxed-*PRF* output $(C, j, \mathbf{T}_{0,j})$.

From relaxed-*PRF*'s output, R can calculate PRF's output.

Thank You ☺

Appendix

Proof of Lemma 1

Proof.

If we define $\mathcal{C}$ from random functions G instead of psuedorandom we get

Proof of Lemma 1

Proof.

If we define $\mathcal{C}$ from random functions G instead of psuedorandom we get

$$\Pr_{C \leftarrow \mathcal{C}} \left[\| C(x) \oplus C(x') \|_H < d \right] = 2^{-n} \sum_{i=0}^{d-1} \binom{n}{i}$$

Proof of Lemma 1

Proof.

If we define $\mathcal{C}$ from random functions G instead of psuedorandom we get

$$\Pr_{C \leftarrow \mathcal{C}} [\|C(x) \oplus C(x')\|_H < d] = 2^{-n} \sum_{i=0}^{d-1} \binom{n}{i}$$

In the case when G is psuedorandom function this probability must be equal to

$$2^{-n} \sum_{i=0}^{d-1} \binom{n}{i} + \vartheta(\kappa)$$

Proof of Lemma 1

Proof.

If we define $\mathcal{C}$ from random functions G instead of psuedorandom we get

$$\Pr_{C \leftarrow \mathcal{C}} \left[\| C(x) \oplus C(x') \|_H < d \right] = 2^{-n} \sum_{i=0}^{d-1} \binom{n}{i}$$

In the case when G is psuedorandom function this probability must be equal to

$$2^{-n} \sum_{i=0}^{d-1} \binom{n}{i} + \vartheta(\kappa)$$

where $\vartheta(\kappa)$ is negligible else we can build a distinguisher $\mathcal{D}$ that distinguishes output of PRF from Random function by calculating hamming weights of inputs. □

The IKNP construction for extending OT with semi-honest receiver

Given a small number of oblivious transfers can we implement a large number of oblivious transfers ?

The IKNP construction for extending OT with semi-honest receiver

Given a small number of oblivious transfers can we implement a large number of oblivious transfers ?
The [IKNP03] construction does that ☺.

The IKNP construction for extending OT with semi-honest receiver

Given a small number of oblivious transfers can we implement a large number of oblivious transfers ?
The [IKNP03] construction does that ☺.
This protocol reduces reduces OT_l^m to OT_m^k which is ultimately reduced to OT_k^k $(m \gg k)$. Overhead is few symmetric key operations.

The IKNP construction for extending OT with semi-honest receiver

- **Input of** S: m pairs $(\mathbf{x}_{j,0}, \mathbf{x}_{j,1})$ of l-bit strings, $1 \leq j \leq m$.

The IKNP construction for extending OT with semi-honest receiver

- **Input of** S: m pairs $(\mathbf{x}_{j,0}, \mathbf{x}_{j,1})$ of l-bit strings, $1 \leq j \leq m$.
- **Input of** R: m selection bits of $\mathbf{r} = (r_1, ...r_m)$.

The IKNP construction for extending OT with semi-honest receiver

- **Input of** S: m pairs $(\mathbf{x}_{j,0}, \mathbf{x}_{j,1})$ of l-bit strings, $1 \leq j \leq m$.
- **Input of** R: m selection bits of $\mathbf{r} = (r_1, ...r_m)$.
- **Common Input**: A security parameter k.

The IKNP construction for extending OT with semi-honest receiver

- **Input of** S: m pairs $(\mathbf{x}_{j,0}, \mathbf{x}_{j,1})$ of l-bit strings, $1 \leq j \leq m$.
- **Input of** R: m selection bits of $\mathbf{r} = (r_1, ...r_m)$.
- **Common Input**: A security parameter k.
- **Oracle**: A random oracle $H : [m] \times \{0,1\}^k \to \{0,1\}^l$.

The IKNP construction for extending OT with semi-honest receiver

- **Input of** S: m pairs $(\mathbf{x}_{j,0}, \mathbf{x}_{j,1})$ of l-bit strings, $1 \leq j \leq m$.
- **Input of** R: m selection bits of $\mathbf{r} = (r_1, ...r_m)$.
- **Common Input**: A security parameter k.
- **Oracle**: A random oracle $H : [m] \times \{0,1\}^k \rightarrow \{0,1\}^l$.
- **Cryptographic Primitive**: An ideal OT_m^k primitive.

The IKNP construction for extending OT with semi-honest receiver

- **Input of** S: m pairs $(\mathbf{x}_{j,0}, \mathbf{x}_{j,1})$ of l-bit strings, $1 \leq j \leq m$.
- **Input of** R: m selection bits of $\mathbf{r} = (r_1, ...r_m)$.
- **Common Input**: A security parameter k.
- **Oracle**: A random oracle $H : [m] \times \{0,1\}^k \rightarrow \{0,1\}^l$.
- **Cryptographic Primitive**: An ideal OT_m^k primitive.

1. S initializes a random vector $\mathbf{s} \in \{0,1\}^k$ and R random $m \times k$ bit matrix T and U such that $\mathbf{T}^i \oplus \mathbf{U}^i = \mathbf{r}$ for $i \in [k]$.

The IKNP construction for extending OT with semi-honest receiver

- **Input of** S: m pairs $(\mathbf{x}_{j,0}, \mathbf{x}_{j,1})$ of l-bit strings, $1 \leq j \leq m$.
- **Input of** R: m selection bits of $\mathbf{r} = (r_1, ...r_m)$.
- **Common Input**: A security parameter k.
- **Oracle**: A random oracle $H : [m] \times \{0,1\}^k \rightarrow \{0,1\}^l$.
- **Cryptographic Primitive**: An ideal OT_m^k primitive.

1. S initializes a random vector $\mathbf{s} \in \{0,1\}^k$ and R random $m \times k$ bit matrix T and U such that $\mathbf{T}^i \oplus \mathbf{U}^i = \mathbf{r}$ for $i \in [k]$.
2. The parties invoke the OT_m^k primitive, where S acts as receiver with input s_i and R as sender with inputs $(\mathbf{T}^i, \mathbf{r} \oplus \mathbf{T}^i)$, $1 \leq i \leq k$.

The IKNP construction for extending OT with semi-honest receiver

3. Let Q denote the $m \times k$ matrix of values received by S.

The IKNP construction for extending OT with semi-honest receiver

3. Let Q denote the $m \times k$ matrix of values received by S.

Remark

$$\mathbf{Q}^i = (s_i \cdot \mathbf{r}) \oplus \mathbf{T}^i \quad (1)$$
$$\mathbf{Q}_j = (r_j \cdot \mathbf{s}) \oplus \mathbf{T}_j \quad (2)$$

The IKNP construction for extending OT with semi-honest receiver

3. Let Q denote the $m \times k$ matrix of values received by S.

Remark

$$\mathbf{Q}^i = (s_i \cdot \mathbf{r}) \oplus \mathbf{T}^i \quad (1)$$
$$\mathbf{Q}_j = (r_j \cdot \mathbf{s}) \oplus \mathbf{T}_j \quad (2)$$

4. S sends $(\mathbf{y}_{j,0}, \mathbf{y}_{j,1})$ where

The IKNP construction for extending OT with semi-honest receiver

3. Let Q denote the $m \times k$ matrix of values received by S.

Remark

$$\mathbf{Q}^i = (s_i \cdot \mathbf{r}) \oplus \mathbf{T}^i \quad (1)$$
$$\mathbf{Q}_j = (r_j \cdot \mathbf{s}) \oplus \mathbf{T}_j \quad (2)$$

4. S sends $(\mathbf{y}_{j,0}, \mathbf{y}_{j,1})$ where

$$\mathbf{y}_{j,0} = \mathbf{x}_{j,0} \oplus H(j, \mathbf{Q}_j)$$
$$\mathbf{y}_{j,1} = \mathbf{x}_{j,1} \oplus H(j, \mathbf{Q}_j \oplus \mathbf{s})$$

The IKNP construction for extending OT with semi-honest receiver

3. For $1 \leq j \leq m$, R outputs $\mathbf{z}_j = \mathbf{y}_{j,r_j} \oplus H(j, \mathbf{T}_j)$

The IKNP construction for extending OT with semi-honest receiver

3. For $1 \leq j \leq m$, R outputs $\mathbf{z}_j = \mathbf{y}_{j,r_j} \oplus H(j, \mathbf{T}_j)$

We can easily verify that $\mathbf{z}_j = \mathbf{x}_{j,r_j}$

The KK13 construction

- If we observe [IKNP03] construction each row(say i^{th}) of $T \oplus U$ is either all 0's or 1's depending on r_i.

The KK13 construction

- If we observe [IKNP03] construction each row(say i^{th}) of $T \oplus U$ is either all 0's or 1's depending on r_i.
- Kolesnikov and Kumersan saw them as repetition code and came up with a similar protocol for extension of $1 - out - of - 2^l\ OT$

The KK13 construction

- If we observe [IKNP03] construction each row(say i^{th}) of $T \oplus U$ is either all 0's or 1's depending on r_i.
- Kolesnikov and Kumersan saw them as repetition code and came up with a similar protocol for extension of $1 - out - of - 2^l$ OT
- Now instead of choice bit r_i receiver has choice string $\mathbf{r_i}$ of length l.

The KK13 construction

- If we observe [IKNP03] construction each row(say i^{th}) of $T \oplus U$ is either all 0's or 1's depending on r_i.
- Kolesnikov and Kumersan saw them as repetition code and came up with a similar protocol for extension of $1 - out - of - 2^l\ OT$
- Now instead of choice bit r_i receiver has choice string $\mathbf{r_i}$ of length l.And let C be an error correcting code of codeword length k.

The KK13 construction

- If we observe [IKNP03] construction each row(say i^{th}) of $T \oplus U$ is either all 0's or 1's depending on r_i.
- Kolesnikov and Kumersan saw them as repetition code and came up with a similar protocol for extension of $1 - out - of - 2^l$ OT
- Now instead of choice bit r_i receiver has choice string $\mathbf{r_i}$ of length l.And let C be an error correcting code of codeword length k.
- The receiver will prepare matrices T and U such that $\mathbf{T}_j \oplus \mathbf{U}_j = C(\mathbf{r_j})$.
- Equation 2 will now become :

$$\mathbf{Q}_j = [C(\mathbf{r_j}) \cdot \mathbf{s}] \oplus \mathbf{T}_j \tag{3}$$

The KK13 construction

- **Input of** S: m tuples $(\mathbf{x}_{j,0}, \ldots \mathbf{x}_{j,n-1})$ of l-bit strings, $1 \leq j \leq m$.

The KK13 construction

- **Input of** S: m tuples $(\mathbf{x}_{j,0}, ...\mathbf{x}_{j,n-1})$ of l-bit strings, $1 \leq j \leq m$.
- **Input of** R: m selection integers of $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m)$ such that $0 \leq \mathbf{r}_j \leq n-1$.

The KK13 construction

- **Input of** S: m tuples $(\mathbf{x}_{j,0}, ...\mathbf{x}_{j,n-1})$ of l-bit strings, $1 \leq j \leq m$.
- **Input of** R: m selection integers of $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m)$ such that $0 \leq \mathbf{r}_j \leq n-1$.
- **Common Input**: a security parameter k and Walsh Hadamard Codes C_{WH}^k.

The KK13 construction

- **Input of** S: m tuples $(\mathbf{x}_{j,0}, ...\mathbf{x}_{j,n-1})$ of l-bit strings, $1 \leq j \leq m$.
- **Input of** R: m selection integers of $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m)$ such that $0 \leq \mathbf{r}_j \leq n - 1$.
- **Common Input**: a security parameter k and Walsh Hadamard Codes C^k_{WH}.
- **Oracle**: a random oracle $H : [m] \times \{0, 1\}^k \rightarrow \{0, 1\}^l$.

The KK13 construction

- **Input of** S: m tuples $(\mathbf{x}_{j,0}, ...\mathbf{x}_{j,n-1})$ of l-bit strings, $1 \leq j \leq m$.
- **Input of** R: m selection integers of $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m)$ such that $0 \leq \mathbf{r}_j \leq n-1$.
- **Common Input**: a security parameter k and Walsh Hadamard Codes C^k_{WH}.
- **Oracle**: a random oracle $H : [m] \times \{0,1\}^k \rightarrow \{0,1\}^l$.
- **Cryptographic Primitive**: An ideal OT^k_m primitive.

The KK13 construction

- **Input of** S: m tuples $(\mathbf{x}_{j,0}, ...\mathbf{x}_{j,n-1})$ of l-bit strings, $1 \leq j \leq m$.
- **Input of** R: m selection integers of $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m)$ such that $0 \leq \mathbf{r}_j \leq n-1$.
- **Common Input**: a security parameter k and Walsh Hadamard Codes C^k_{WH}.
- **Oracle**: a random oracle $H : [m] \times \{0,1\}^k \rightarrow \{0,1\}^l$.
- **Cryptographic Primitive**: An ideal OT^k_m primitive.

1. S initializes a random vector $\mathbf{s} \in \{0,1\}^k$ and R random $m \times k$ bit matrix T and U such that $\mathbf{T}_j \oplus \mathbf{U}_j = C(\mathbf{r_j})$.

The KK13 construction

- **Input of** S: m tuples $(\mathbf{x}_{j,0}, ...\mathbf{x}_{j,n-1})$ of l-bit strings, $1 \leq j \leq m$.
- **Input of** R: m selection integers of $\mathbf{r} = (\mathbf{r}_1, ...\mathbf{r}_m)$ such that $0 \leq \mathbf{r}_j \leq n-1$.
- **Common Input**: a security parameter k and Walsh Hadamard Codes C^k_{WH}.
- **Oracle**: a random oracle $H : [m] \times \{0,1\}^k \rightarrow \{0,1\}^l$.
- **Cryptographic Primitive**: An ideal OT^k_m primitive.

1. S initializes a random vector $\mathbf{s} \in \{0,1\}^k$ and R random $m \times k$ bit matrix T and U such that $\mathbf{T}_j \oplus \mathbf{U}_j = C(\mathbf{r_j})$.
2. The parties invoke the OT^k_m primitive, where S acts as receiver with input s_i and R as sender with inputs $(\mathbf{T}^i, \mathbf{U}^i)$, $1 \leq i \leq k$,

The KK13 construction

3. Let Q denote the $m \times k$ matrix of values received by S.

The KK13 construction

3. Let Q denote the $m \times k$ matrix of values received by S.

Remark

$$\mathbf{Q}_j = (C(\mathbf{r_j}) \cdot \mathbf{s}) \oplus \mathbf{T}_j$$

The KK13 construction

3. Let Q denote the $m \times k$ matrix of values received by S.

Remark

$$\mathbf{Q}_j = (C(\mathbf{r_j}) \cdot \mathbf{s}) \oplus \mathbf{T}_j$$

4. S sends $(\mathbf{y}_{j,0}, ...\mathbf{y}_{j,n-1}$ *for* $j \in [m]$ *and* $r \in [n-1])$ where

The KK13 construction

3. Let Q denote the $m \times k$ matrix of values received by S.

Remark

$$\mathbf{Q}_j = (C(\mathbf{r_j}) \cdot \mathbf{s}) \oplus \mathbf{T}_j$$

4. S sends $(\mathbf{y}_{j,0}, \ldots .\mathbf{y}_{j,n-1}$ *for* $j \in [m]$ *and* $r \in [n-1])$ where

$$\mathbf{y}_{j,r} = \mathbf{x}_{j,r} \oplus H(j||\mathbf{Q}_j \oplus (\mathbf{C}(\mathbf{r}) \cdot \mathbf{s})$$

5. For $1 \leq j \leq m$, R outputs $\mathbf{z}_j = \mathbf{y}_{j,\mathbf{r}_j} \oplus H(j, \mathbf{T}_j)$

The KK13 construction

3. Let Q denote the $m \times k$ matrix of values received by S.

Remark

$$\mathbf{Q}_j = (C(\mathbf{r_j}) \cdot \mathbf{s}) \oplus \mathbf{T}_j$$

4. S sends $(\mathbf{y}_{j,0}, \ldots .\mathbf{y}_{j,n-1}$ *for* $j \in [m]$ *and* $r \in [n-1])$ where

$$\mathbf{y}_{j,r} = \mathbf{x}_{j,r} \oplus H(j||\mathbf{Q}_j \oplus (\mathbf{C}(\mathbf{r}) \cdot \mathbf{s})$$

5. For $1 \leq j \leq m$, R outputs $\mathbf{z}_j = \mathbf{y}_{j,\mathbf{r}_j} \oplus H(j, \mathbf{T}_j)$
6. It is easy to verify that R can't learn any other value and is able to learn value corresponding to his choice integer only.

Proof of lemma 2

Proof.

In the $m - RK - PRF$ game with the defined PRF, we can write adversary's(controlling R) view as :

$$\left(C, \{\mathbf{T}_{0,j}\}_{j\in[m]}, \{H(p_i||\mathbf{T}_{0,p_i} \oplus [(C(\mathbf{x}_{p_i}) \oplus C(\mathbf{y}_i)) \cdot \mathbf{s}])\}_{i\in[m]}\right) \tag{4}$$

Proof of lemma 2

Proof.

In the $m - RK - PRF$ game with the defined PRF, we can write adversary's(controlling R) view as :

$$\left(C, \{\mathbf{T}_{0,j}\}_{j\in[m]}, \{H(p_i||\mathbf{T}_{0,p_i} \oplus [(C(\mathbf{x}_{p_i}) \oplus C(\mathbf{y}_i)) \cdot \mathbf{s}])\}_{i\in[m]}\right) \quad (4)$$

Now, in the game adversary chooses m strings y_i. So we have m terms of the form $C(x_{p_i}) \oplus C(y_i)$ for $x_{p_i} \neq y_i$.

Proof of lemma 2

Proof.

In the $m - RK - PRF$ game with the defined PRF, we can write adversary's(controlling R) view as :

$$\left(C, \{\mathbf{T}_{0,j}\}_{j\in[m]}, \{H(p_i||\mathbf{T}_{0,p_i} \oplus [(C(\mathbf{x}_{p_i}) \oplus C(\mathbf{y}_i)) \cdot \mathbf{s}])\}_{i\in[m]}\right) \tag{4}$$

Now, in the game adversary chooses m strings y_i. So we have m terms of the form $C(x_{p_i}) \oplus C(y_i)$ for $x_{p_i} \neq y_i$. By our choice of C each of these have hamming weight atleast d with probability atleast $1 - \frac{2^{-\epsilon}}{m}$.

Proof of lemma 2

Proof.

In the $m - RK - PRF$ game with the defined PRF, we can write adversary's(controlling R) view as :

$$\left(C, \{\mathbf{T}_{0,j}\}_{j\in[m]}, \{H(p_i||\mathbf{T}_{0,p_i} \oplus [(C(\mathbf{x}_{p_i}) \oplus C(\mathbf{y}_i)) \cdot \mathbf{s}])\}_{i\in[m]}\right) \tag{4}$$

Now, in the game adversary chooses m strings y_i. So we have m terms of the form $C(x_{p_i}) \oplus C(y_i)$ for $x_{p_i} \neq y_i$. By our choice of C each of these have hamming weight atleast d with probability atleast $1 - \frac{2^{-\epsilon}}{m}$.

Let A_i be the event when $i^t h$ such term has hamming weight less than d. Then :

Proof of lemma 2

Proof.

In the $m - RK - PRF$ game with the defined PRF, we can write adversary's(controlling R) view as :

$$\left(C, \{\mathbf{T}_{0,j}\}_{j\in[m]}, \{H(p_i||\mathbf{T}_{0,p_i} \oplus [(C(\mathbf{x}_{p_i}) \oplus C(\mathbf{y}_i)) \cdot \mathbf{s}])\}_{i\in[m]}\right) \qquad (4)$$

Now, in the game adversary chooses m strings y_i. So we have m terms of the form $C(x_{p_i}) \oplus C(y_i)$ for $x_{p_i} \neq y_i$. By our choice of C each of these have hamming weight atleast d with probability atleast $1 - \frac{2^{-\epsilon}}{m}$.
Let A_i be the event when $i^t h$ such term has hamming weight less than d. Then :

$$\Pr\left[A_1^\complement \cap A_2^\complement ..... \cap A_m^\complement\right] = 1 - \Pr\left[(A_1^\complement \cap A_2^\complement ..... \cap A_m^\complement)^\complement\right]$$
$$= 1 - \Pr\left[A_1 \cup A_2 ..... \cup A_m\right]$$

Proof Continued.

By union bound

$$\geq 1 - m\frac{2^{-\epsilon}}{m} = 1 - 2^{-\epsilon}.$$

Proof Continued.

By union bound

$$\geq 1 - m\frac{2^{-\epsilon}}{m} = 1 - 2^{-\epsilon}.$$

So, by above calculation all such terms have hamming weight atleast d with overwhelming probability$(1 - 2^{-\epsilon})$.

Proof Continued.

By union bound

$$\geq 1 - m\frac{2^{-\epsilon}}{m} = 1 - 2^{-\epsilon}.$$

So, by above calculation all such terms have hamming weight atleast d with overwhelming probability$(1 - 2^{-\epsilon})$.
Conditioning on this event apply $d-$ hamming correlation robust property of H to conclude that H outputs are indistinguishable from random. $\square$

References

Łukasz Chmielewski and Jaap-Henk Hoepman.
Fuzzy private matching (extended abstract).
Cryptology ePrint Archive, Report 2007/363, 2007.
http://eprint.iacr.org/2007/363.

Jan Camenisch, Gregory Neven, and Abhi Shelat.
Simulatable adaptive oblivious transfer.
EUROCRYPT 2007, 2007.

Michael J. Freedman, Yuval Ishai, Benny Pinkas, and Omer Reingold.
Keyword search and oblivious pseudorandom functions, 2005.

Carmit Hazay and Yehuda Lindell.
Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries.
J. Cryptol., 23(3):422–456, July 2010.

Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank.
Extending oblivious transfers efficiently.
CRYPTO, 2003.

Vladimir Kolesnikov, Ranjit Kumaresan, Mike Rosulek, and Ni Trieu.
Efficient batched oblivious prf with applications to private set intersection.
Cryptology ePrint Archive, Report 2016/799, 2016.
http://eprint.iacr.org/2016/799.

Benny Pinkas, Thomas Schneider, Gil Segev, and Michael Zohner.
Phasing: Private set intersection using permutation-based hashing.
Cryptology ePrint Archive, Report 2015/634, 2015.
http://eprint.iacr.org/2015/634.