Fully Distributive Evaluation Protocol: Pseudocode

Vivek Sharma

Jan 11, 2020

1 Introduction

This document contains four parts. First part includes the introduction to the protocol. Second part contains the pseudocode for the protocol and sub protocol. Third part includes a dummy example at some imaginary j^{th} iteration. The last part contains a list of questions regarding the protocol.

The protocol is divided into three phases. The middle phase is an interactive phase running a sub protocol $\pi_{2,3}$ which takes input from three servers, which is their additive secret share mod 2, and outputs two additive share mod 3 shared by two servers. The first and last phase are non-interactive, meaning, they can be computed locally.

1.1 Phase 1: NonInteractive computation of Additive share by each Server

Each server S_i holds replicated additive shares of key $k_i \in \mathbb{Z}_n^2$ and $x_i \in \mathbb{Z}_n^2$ and computes $h_i \in \mathbb{Z}_2^m$, which is the multiplication of key and input over $\mathbb{Z}_2$. This computation is performed locally.

1.2 Phase 2: Interactive computation of π_{23} protocol

1. Each server, at this point have locally computed their shares, which was the multiplication of two vectors.

2. Server 1 randomly chooses a value $c \in \mathbb{Z}_3^m$ and each bit of value is converted to it's 2-bit representation to form c_0 and c_1 respectively.

3. Meanwhile, Server 1, 2 and 3 runs sub-protocol for m instances(m is the length of additive share and also the value of c, which is with server 1.

   For $1 \leq j \leq m$:

   Each server $s_i, i \in 1, 2, 3$ share their input $h_{i,j}$ [Note: $h_{i,j}$ is the input of server s_i in j^{th} iteration]

   Compute combined XOR of their input: $comb := h_1 \oplus h_2 \oplus h_3$

Multiply one part of c(c_0), with comb and other part (c_1) with $\neg comb$ and XOR both the result, this forms d_0.

To compute d_1, XOR the c_0 and $\neg c_1$, and multiply the result with the XOR of secret share of the servers.

The final result $d = d_0, d_1 \in \{0,1\}^2$ is converted back into $\mathbb{Z}_3$

At the end of this phase, Server 1 has $c \in \{0,1\}^m$ and Server 2 has received the output $d \in \mathbb{Z}_3$. The combination of values with Server one and two (i.e. c and d) yields the additive mod 3 of the secret share of the inputs by the Servers. Mathematically $c+d = h_1+h_2+h_3(mod 3)$

1.3 Phase 3: Non Interactive evaluation of function map by Server 1 and Server 2

The Servers in possession of random value c and the interactively computed value d, apply map_G function on their input and compute their share in $\mathbb{Z}_3$. Note: This map_G is guessed to be the additive mod G function, we saw at the implementation of weak PRF.

2 Pseudocode

2.1 piprotocol():

1. Server S_i : $h_i := k \cdot x \pmod 2 \in \mathbb{Z}_2^m$. //matrix vector multiplication
2. S_1 : Selects c $\xleftarrow{\text{R}} \mathbb{Z}_3^m$. //randomly selected from the field.
3. for $1 \leq j \leq m$: //length of c and h_i
4. Server S_i : $\pi_{2,3}(c_0, c_1, h_1, h_2, h_3)$ //All servers run parallel instances of the protocol
5. S_1: $\sum_{1 \leq n \leq m} c_n \pmod G$ and S_2: $\sum_{1 \leq n \leq m} d_n \pmod G$

2.2 $\pi_{2,3}(c_0, c_1, h_1, h_2, h_3)$:

1. $comb := h_1 \oplus h_2 \oplus h_3$ //secret share of each server at $j^{th} iteration$
2. $d_0 = c_0 \cdot comb \oplus c_1 \cdot \neg comb$
3. $d_1 = c_0 \oplus \neg c_1 \cdot comb$
4. return d_0 and d_1 //Server two obtains d by combining d_0 and d_1

3 Example:

- Say at j^{th} instance, $h_1 = 1, h_2 = 0, h_3 = 1$ and server randomly chooses c = 2, so $c_0 = 1, c_1 = 0$
- $comb = 0, andd_0 = 0, d_1 = 0$ as computed by formula given above.
- This satisfies the formula $c + d = h_1 \oplus h_2 \oplus h_3 \pmod 3$, is true in this case.
- Server one and two apply map_G function on their value $candd \in \mathbb{Z}_3$

4 Questions:

1. What is it for? The description says that it is for distributed PRF evaluation? What are the possible evaluation where the secret shared between three servers is then modified to be between two server.
2. Below Figure 1: Server S_i knows a_j and b_j (which is k_i and x_i), isn't it private key and private input of a user which must remain private. I can understand h_i being shared, the other server won't be able to track back the value of k and x, but the concept of replicated secret sharing confuses me.
3. Does all the server compute subprotocol $\pi_{2,3}$, if yes, do they all get the random value c selected by Server one and the shares of other input h_i, and why do we need replicated secret sharing scheme if all we need is h_i
4. How does it happen that only server two obtains $c^{'}$ (or the value d_0 and d_1)
5. Is the map_G function same as the one defined in weak PRF as a public matrix. There it is mentioned that the use of map function is to elongate the size of single dimensional output of PRF, is it true here too? Is it necessary in distributed PRF evaluation setting.