Weak PRF Protocol: Pseudocode

Vivek Sharma

Jan 11, 2020

1 Fully Distributive Evaluation Protocol

The key is structured as a vector in $\mathbb{Z}_2$

The protocol is divided into three phases:

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: $comb := h_1 \oplus h_2 \oplus h_{13}$

   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 $\mathbb{Z}_3$

2 Pseudocode

piprotocol

1. Server S_i : $h_i := k.x \pmod 2 \in \mathbb{Z}_2^m$.
2. S_1 : Selects c $\xleftarrow{R} \mathbb{Z}_3^m$.
3. for $1 \leq j \leq m$:
4. Server S_i : $\pi_{2,3}()$ //All servers run parallel instances of the protocol
5. S_1: $\sum_{1 \leq n \leq m} c_n \pmod G$ and S_2: $\sum_{1 \leq n \leq m} d_n \pmod G$

3 Explanation:

The protocol consists of 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.

4 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 $c + d = h_1 + h_2 + h_3(mod3)$, is true in this case.
- Server one and two apply map_G function on their value $candd \in \mathbb{Z}_3$