Yao's Garbled Circuit Optimizations: Free XOR, Point and Permute, Half Gates

Lillian Bernstein and Ria Rajesh

May 22, 2024

Yao's Garbled Circuit is a circuit of gates including XOR, AND, and NOT gates. The garbled circuit includes a vector of garbled gates and two vectors of garbled wires, with the gates containing entries of ciphertexts. One wire vector contains wire labels for zero input values and the other wire vector contains wire labels for one input values. The garbler loops through each gate, with the inputs to the ciphertexts varying depending on the type of gate. The garbler sends the entire garbled circuit of garbled gates and garbled wires to the evaluator. The evaluator receives their own input via Oblivious Transfer and loops through the garbled circuit they received to arrive at a final output.

Our first optimization technique is Free XOR. Free XOR is an optimization procedure for Yao's Garbled Circuit to reduce the need for the garbler to generate and send ciphertexts for any XOR gate. The second optimization is point and permute. Point and permute removes the need for the evaluator to check and decrypt all ciphertexts by associating each label with a choice bit p_a and p_b. The evaluator can use these choice bits to figure out which of the ciphertexts sent is the correct one to decrypt.

Our final optimization was Half Gates, which is the core of the capstone. Half Gates is a technique that makes it so that we only require two ciphertexts per garbled AND gate. It uses the fact that

$$a \wedge b = (a \wedge (b \oplus r)) \oplus (a \wedge r)$$

for the random select bit r. The two sides of the middle $\oplus$ represent the two half gates. For each half gate there is a ciphertext, with one referred to as the Evaluator's ciphertext and the other referred to as the Garbler's ciphertext. The Garbler creates both of them as such:

$$T_G = H(A_0) \oplus H(A_1) \oplus p_a R$$

$$T_E = H(B_0) \oplus H(B_1) \oplus A_0$$

Here, A_i and B_i refer to the left and right side labels for inputs $i \in \{0, 1\}$, p_i refers to the choice bit for $i \in \{a, b\}$. The number R is the same as Δ from FreeXOR, so $A_1 = A_0 \oplus R$ and so on. In our implementation, the hash H is SHA256. The Garbler uses the ciphertexts to generate the labels for the output wires for the garbled circuit as such:

$$W_G^0 = H(A_0) \oplus p_a T_G$$

$$W_E^0 = H(B_0) \oplus p_a(T_E \oplus A_0)$$

$$W^0 = W_G^0 \oplus W_E^0$$

$$W^1 = W^0 \oplus R$$

After generating the labels, the Garbler sends the Evaluator the two ciphertexts T_G and T_E. Then, the evaluator performs the following computations to extract the proper wire:

$$W_G = H(W_a) \oplus s_a(T_G)$$

$$W_E = H(W_b) \oplus s_b(T_E \oplus W_a)$$

With s_i being the choice bits for $i \in \{0, 1\}$, and W_a and W_b being the left and right side wires of the given gate. The output wire is computed as such:

$$W = W_G \oplus W_E$$