Some Thoughts of Implementing AND Gates

1 Considering min function

1.1 General idea

I came up with this idea from the truth:

$$0 \wedge 1 = \mathsf{min}(0, 1).$$

Thus I wonder if I can exploit such quality to implement a AND gate. Considering we map each boolean value v_i ($v_i \in \{\texttt{TRUE}, \texttt{FALSE}\}$) to a L-bit wire label W_i ($W_i \in [0, 2^{L-1}]$). Now the generator $\mathcal{G}$ will differentiate label representing TRUE from that representing FALSE via a function $v_i = \mathsf{decode}(W_i,\ t)$, where $t \in [0, 2^{L-1}]$ is a random secret chosed by $\mathcal{G}$:

$$\mathsf{decode}(W_i, t) = \begin{cases} \texttt{TRUE} & W_i <= t, \\ \texttt{FALSE} & W_i > t. \end{cases}$$

Therefore, as an evaluator $\mathcal{E}$, it will evaluate an AND gate via a function $W_k = \mathsf{eval}(W_i,\ W_j)$:

$$\mathsf{eval}(W_i, W_j) = \mathsf{min}(W_i, W_j) = W_k.$$

At the end of computation, $\mathcal{E}$ will learn the real value (i.e., TRUE or FALSE) by invoking $\mathsf{decode}(\cdot,\ t)$, in which t is received from generator $\mathcal{G}$. Moreover, OR gates can also be implemented in this way by replacing min with max.

1.2 Defects

It is not secure statistically since W_i with lower value has much more possiblity to be `TRUE`.

2 Considering random bit b

2.1 General idea

Since above scheme is not secure, I consider using a random bit b_i to ensure security of wire label W_i. Now the threshold $t = 2^{L-2}$ is a common value between both parties. And the generator $\mathcal{G}$ defines function $v_i = \mathsf{decode}(W_i, b_i)$ as:

$$\mathsf{decode}(W_i, b_i) = \begin{cases} \texttt{TRUE} & W_i <= t \text{ and } b_i == 0, \\ \texttt{FALSE} & W_i > t \text{ and } b_i == 0, \\ \texttt{FALSE} & W_i <= t \text{ and } b_i == 1, \\ \texttt{TRUE} & W_i > t \text{ and } b_i == 1, \end{cases}$$

where $b_i \in \{0, 1\}$ is chosed randomly by $\mathcal{G}$ for each wire.

This scheme indeed behaves equivalently to the scheme with $W_i = b_i \oplus v_i$. If we evaluate the latter as the evaluator $\mathcal{E}$, we will find that:

$$\begin{aligned} W_i \wedge W_j &= (b_i \oplus v_i) \wedge (b_j \oplus v_j) \\ &= (b_i \wedge b_j) \oplus (v_i \wedge v_j) \oplus (b_i \wedge v_j) \oplus (b_j \wedge v_i) \\ &= (b_i \wedge b_j) \oplus (v_i \wedge v_j) \oplus (b_i \wedge W_j) \oplus (b_j \wedge W_i) \\ &= W_k \end{aligned}$$

For generator $\mathcal{G}$, he cannot determine the b_k for W_k (since he does not know W_i and W_j). Therefore, the evaluator $\mathcal{E}$ will not be able to decode the truth value at the end.

3 Why are these designs impractical

Yes, I did read the paper published on Eurocrypt'15: *Two Halves Make a Whole: Reducing Data Transfer in Garbled Circuits using Half Gates*, in which it proved that two entries is the optimal solution for AND gates implementation.

However, before reading this paper, my gut instinct has told me that evaluate a *secure* AND/OR gate for free is some kind of impractical. For a *secure* gate, the evaluator $\mathcal{E}$ is not able to guess the true value $v_i (v_j)$ from each input wire $W_i(W_j)$ respectively (i.e., the possibility of its correct guess is supposed to be nearly $\frac{1}{2}$). Suppose there is a secure AND gate and $\mathcal{E}$ can evaluate it for free, then the possibility of output wire $W_k == \texttt{TRUE}$ is nearly $\frac{1}{4}$. It's not balanced. Therefore these 'unbalenced' gates (i.e. AND, OR, NAND, NOR, etc.) cannot evaluated both secure and free.