NSUCRYPTO2024

Problem 5: Reverse engineering

October 21, 2024

Solution

The function

$$f_{2n}\left(x_1,\ldots,x_{2n}\right)=\bigoplus_{i=1}^{n} x_i x_{i+n} \prod_{j=i+1}^{n}\left(x_j \oplus x_{j+n}\right)$$

models the addition of two n-bit integers a and b, where x_1 and x_n denote the least significant and most significant bits of a, respectively, and x_{n+1} and x_{2n} represent the least significant and most significant bits of b. If the result of this addition exceeds n bits (i.e., an overflow occurs), the function returns 1; otherwise, it returns 0.

Proof. We begin by breaking down the given function f_{2n}:

$$\begin{aligned}
f_{2n} &= \left(x_1 x_{1+n}\right)\left(x_2 \oplus x_{2+n}\right)\cdots\left(x_n \oplus x_{2n}\right) \oplus \cdots \oplus \left(x_{n-1} x_{2n-1}\right)\left(x_n \oplus x_{2n}\right) \oplus x_n x_{2n} \\
&= \left[\left(x_1 x_{1+n}\right)\left(x_2 \oplus x_{2+n}\right)\cdots\left(x_{n-1} \oplus x_{2n-1}\right) \oplus \cdots \oplus \left(x_{n-1} \oplus x_{2n-1}\right)\right]\left(x_n \oplus x_{2n}\right) \oplus x_n x_{2n} \\
&= f_{2(n-1)}\left(x_n \oplus x_{2n}\right) \oplus x_n x_{2n}
\end{aligned}$$

We start examining the base case where $n=1$. In this case, we are adding two 1-bit integers. The function $f_2(x_1,x_2)$ determines whether the sum results in an overflow. Specifically, we have:

$$f_2(x_1,x_2)=\begin{cases} 1 & \text{if } x_1=x_2=1, \\ 0 & \text{otherwise.} \end{cases}$$

This represents the base case where an overflow occurs when both bits are 1.

Next, we move to the inductive case for two k-bit integers. Assume that the function f_{2k} correctly identifies overflow for two k-bit integers. We now need to demonstrate that $f_{2(k+1)}$ correctly identifies overflow for two $(k+1)$-bit integers a and b.

For the inductive step, consider the addition of the two $(k+1)$-bit integers $a=(x_1,\ldots,x_{k+1})$ and $b=(x_{k+2},\ldots,x_{2k+2})$. The function $f_{2(k+1)}$ can be defined recursively as:

$$f_{2(k+1)} = f_{2k}\left(x_{k+1} \oplus x_{2k+2}\right) \oplus \left(x_{k+1} x_{2k+2}\right),$$

where x_{k+1} and x_{2k+2} represent the most significant bits of a and b, respectively.

If the two most significant bits of a and b are both 1, then $f_{2(k+1)} = f_{2k} \cdot 0 \oplus 1$, indicating an overflow due to the sum of the two most significant bits. Otherwise, f_{2k} correctly handles any overflow from the sum of the k least significant bits, effectively acting as a carry.

Thus, $f_{2(k+1)}$ detects overflow by accounting for both the sum of the k least significant bits (handled by f_{2k}) and the overflow arising from the addition of the two most significant bits. $\square$

Please refer to the solution script for more details on NSUCRYPTO2024 Problem 5.