**QUESTIONS DISCUSSED ON THE MEETING**

**1) What is Hamming Distance? How is it used to measure the Quality of a PUF?**

Hamming distance is used to define some essential notations in coding theory, such as error detecting and error correcting codes.

In a case where code c is said to be K-errors detecting if any two codewords c1 and c2 from c that have a hamming distance less than k coincide, otherwise code is k-errors detecting if and only if the minimum hamming distance between any two of its codewords is at least k+1(i.e) Difference in two codewords is measured in terms of Hamming Distance.

Hamming Weight is defined as the “Number of Non-Zero Entries”. It’s used for the number of bit flips needed.

For the **Anderson’s 128-bit PUF the average hamming weight should be 64 bits to have a good distortion so that it would be hard to break the challenge.**

**2) Understanding” Measuring intra distance of roughly 90 bits, corresponds to an error of 30% (Section III B) “**

Intra-Distance is defined as the number for bits in a PUF response that change when the same challenge input is given to the PUF. Usually, this happens due to environmental variation and statistical noise.

When measuring the Anderson’s PUF for roughly say 90 bits, it gives different values for the same challenge in Anderson’s PUF. So, this is due to the change in glitch value doesn’t have sufficient time to show up. For this we have two methods to overcome the short delay time so that we can have a clear PUF output.

First Method:

by adding extra multiplexers in between the carry chains.

Second Method:

Measurement after specific delay

**3) How will you measure the Quality of modified Anderson PUF using Hamming Distance Metric?**

We need to measure the same way as Anderson did for his PUF design using Hamming Weight.

In this method, we will check the quality of PUF and by checking the Hamming Distances with an intra distance of about 90 bits. It should be on an average of around 64 bits for a 128 bit PUF.

**4) Clear Schematic of Modified Anderson PUF.**

A screenshot of a cell phone

Description generated with high confidence

We have two LUT’s that generate the opposite patterns of sequence such that as follows.

LUT A:0101010010101010(0\*5555)

LUT B:1010101010101010(0\*AAAA)

These are the initial values for the LUT’s. The IN pin makes sure that this pattern continues every 16 cycles for the 2 LUT’s and the OUT pins are used to drive the select input pins on carry chain multiplexers. Both carry multiplexer have their ‘0’ input tied to logic-1 and the bottom carry chain multiplexer has ‘1’ data input tied to logic-0.

Consider the initial case LUTA is at logic-0 so we get N2 at logic-0. The output pin of LUT B is 1. At rising edge, the OUTPUT pin of LUTA will change from logic-0 to logic-1 and the output pin of LUT B will change from logic-1 to logic-0. Although LUT A and the multiplexer it drives as same as the LUT B, the two pieces of circuitry in fact experience different delays due to random process variations.

Two cases will be looked upon based on the random process variation of the circuit:

First Case: when LUT B and the multiplexer is faster than LUT A and its multiplexer. In this case, LUT B transitions from logic 1 to 0, similarly N1 also transitions from logic 0 to 1. Following that slower LUT A transitions from logic 0 to 1,such that signal N2 is kept constant at logic 1 throughout the process.

Second Case: when LUT A and its multiplexer is faster than LUT B and its multiplexer. In this case, LUT A ‘s OUT pin transitions from logic 0 to 1 and net N1 has not yet transitioned from logic 1 to logic 0. Therefore, we have a short negative spike(glitch) will appear on N2 for the period before N1 transitions to logic-1.

REFERENCES:

1.https://ipfs.io/ipfs/QmXoypizjW3WknFiJnKLwHCnL72vedxjQkDDP1mXWo6uco/wiki/Hamming\_weight.html