

ZK Circuits VS FHE Circuits

A comparison of the programming model behind
zk circuits and FHE circuits



Difference: Constraints VS circuit

- (most) zk proof systems: set of constraints.
 - Can use non-determinism
 - Example: R1CS, CCS, Plonkish, AIR
- 1st: Compute a solution (witness)
- 2nd: Verify the solution with constraints
 - Example: $c = a / b$
 - Constraint: $b * c == a$



Difference: Constraints VS circuit

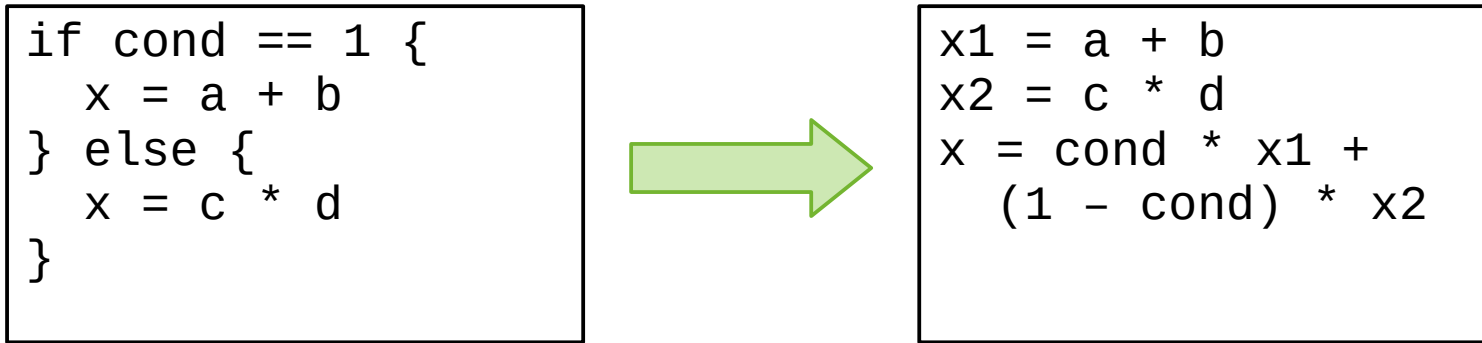
- MPC & FHE [1]: Arithmetic/boolean circuit
 - Example: Garbled Circuits, FHEW, Shamir Secret Sharing MPC
- 1st: Define the function as an arithmetic/boolean circuit
- 2nd: Evaluate the circuit from the inputs
 - Example: $a = (b \text{ AND } c) \text{ OR } d$
 - Evaluation: $\text{tmp} = b \text{ AND } c$; $a = \text{tmp OR } d$

[1] And some zk proof systems like GKR



Similarity: No branching [1]

- Encoding branches means taking both paths and then unifying
- Example:

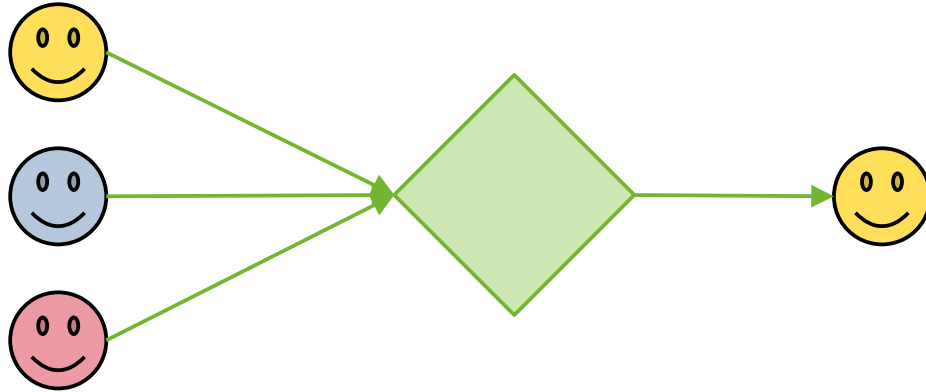


[1] Unless you have a zkVM



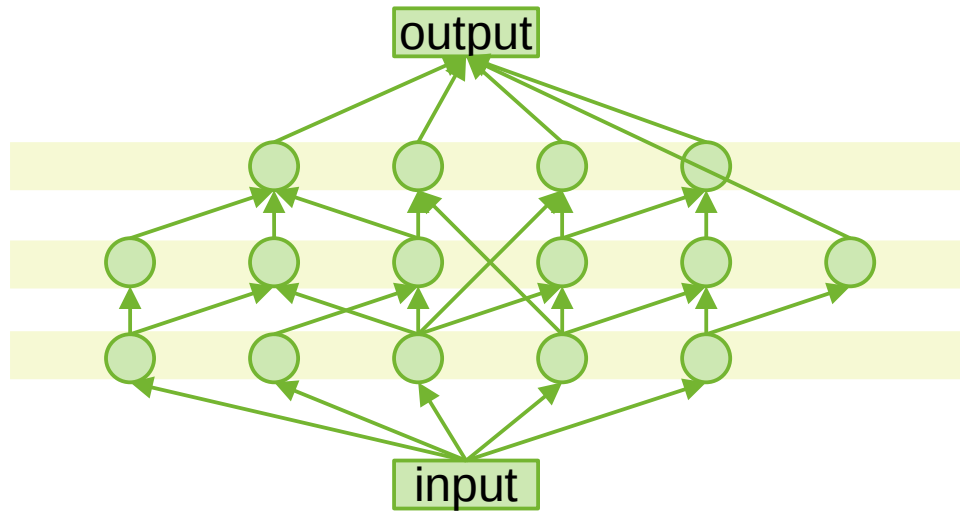
Difference: private data ownership

- zk: Single private data source
 - Public & private
- MPC/mpFHE: Multiple users private data
 - Public & private(for whom?)



Difference: Parallelism

- ZK: parallelism at the proof system level
- FHE/MPC: parallelism at the circuit level




Each layer is
evaluated in
parallel



Difference: Fields VS booleans

- (many) zk proof systems use field arithmetic
- FHE/MPC can work with boolean circuits
- Boolean logic:
 - expensive in zk
 - cheap in FHE/MPC
- u8, u16, u32:
 - expensive in zk
 - cheap in FHE/MPC



-1 ==
218882428718392752222
464057452572750886963
111572978236626890378
94645226208582

-1 == 0xffff



Difference: circuit compilers

- ZK: quite mature, lots of options, good integration
 - circom, noir, RISCv zkVMs, PIL, Lurk, Leo, o1js, Cairo
- FHE/MPC: less mature, few options, non-reusable
 - MP-SPDZ, concrete, HEIR, google/fully-homomorphic-encryption, summon



The End

