Generic verifiable computation and zero knowledge proof

Verifiable Computation (VC)

server

client

digest δ

function f

result = f(data) + 

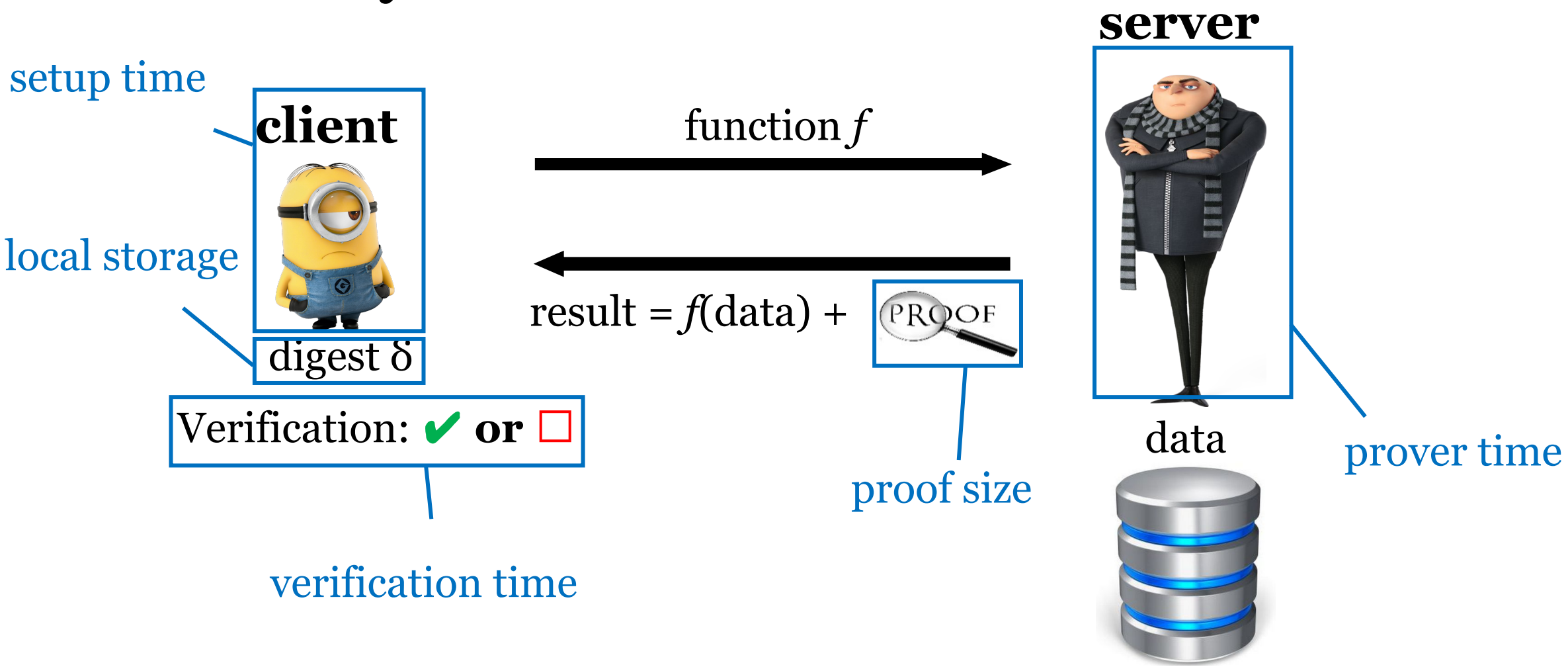
data

Verification: ✔ **or** ☐

Correctness/completeness: Pr[result = f(data) and proof is honest and verification is ✓] = 1

Soundness/security: Pr [result ≠ f (data) and verification is ✓] ≤ $\frac{1}{2^{100}}$

Efficiency measures



Generic VC

- Model functions as arithmetic circuits

SNARK

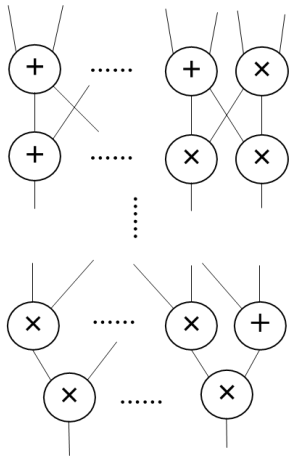
- Succinct Non-interactive ARgument of Knowledge

✔ Supports all functions (modeled as arithmetic circuit)

✔ Constant proof size

✔ Fast verification time

SNARK



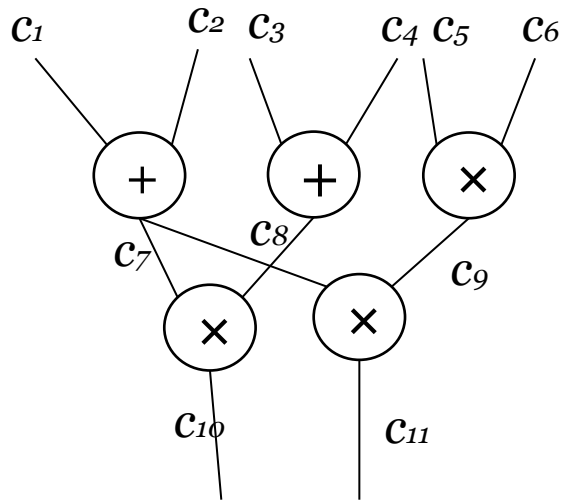
client

result = *C*(data) + PROOF

server

data

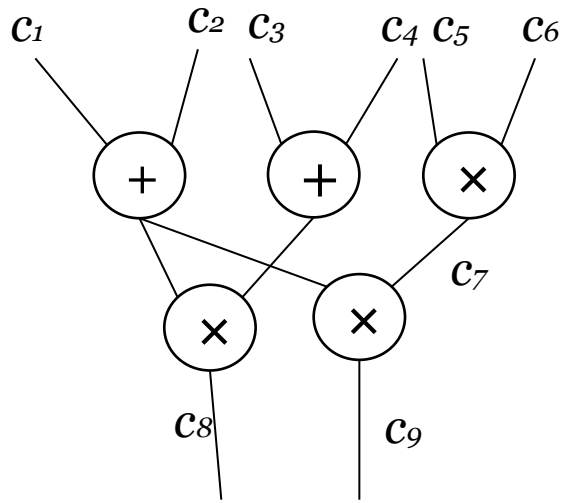
Satisfying assignment of circuits



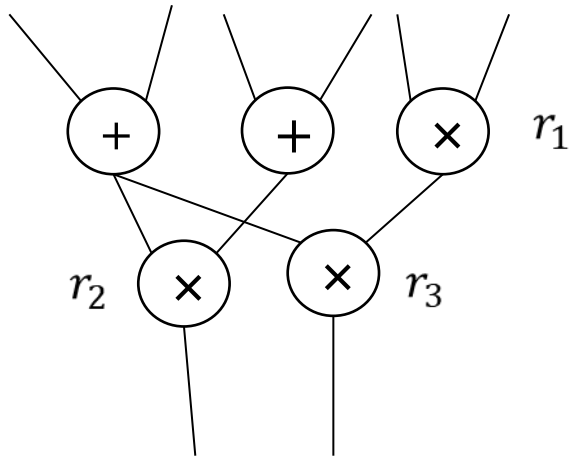
Proving C(data) = output $\rightarrow (c_1, c_2, \ldots c_{11})$ with conditions defined by the circuit

Verifying is easier than computing

Labeling of wires and gates

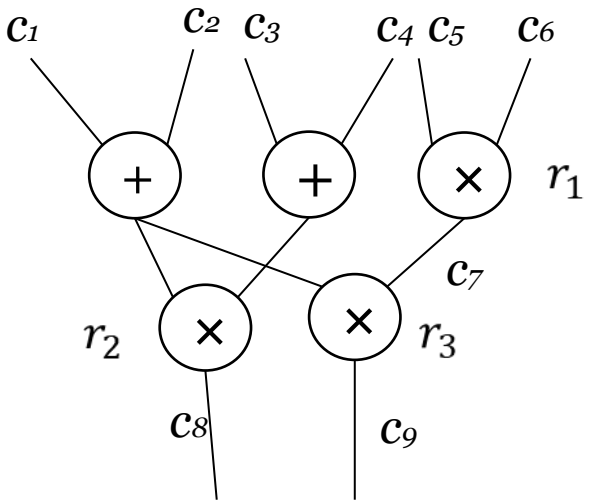


Labeling of wires and gates

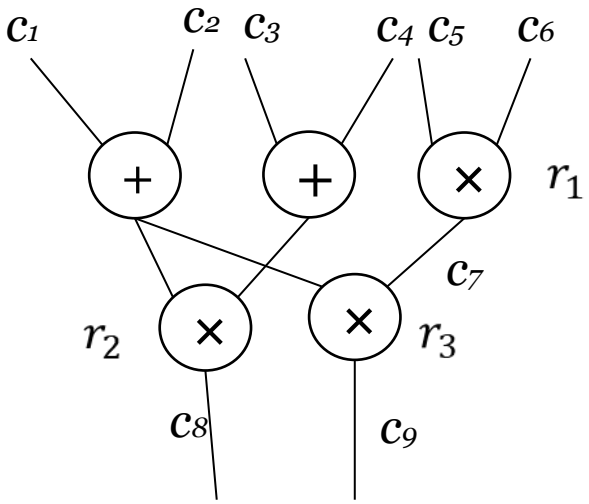


Assign a different number to each multiplication gate

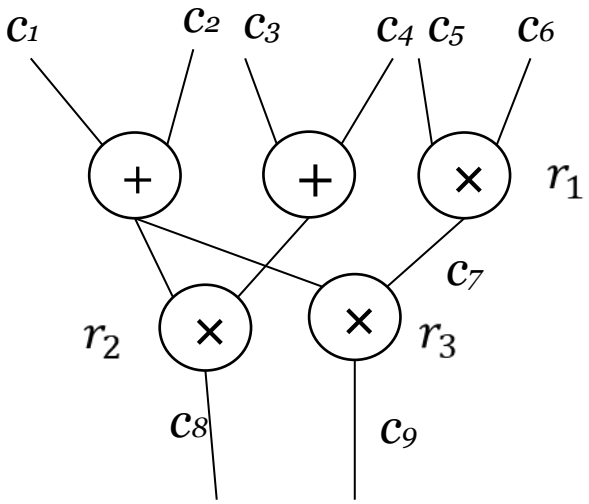
Encoding circuits to polynomials



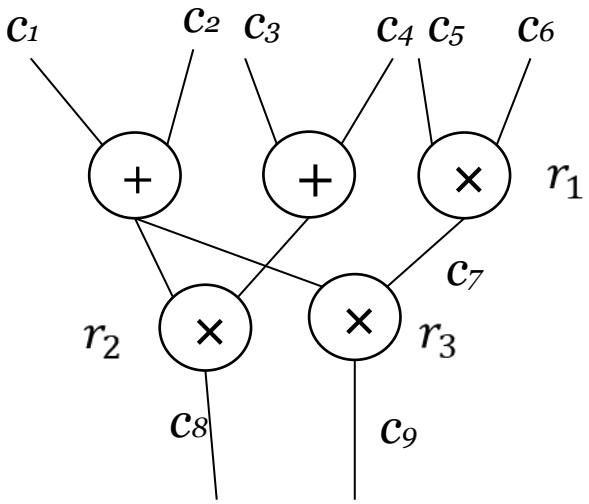
Encoding circuits to polynomials



Encoding circuits to polynomials

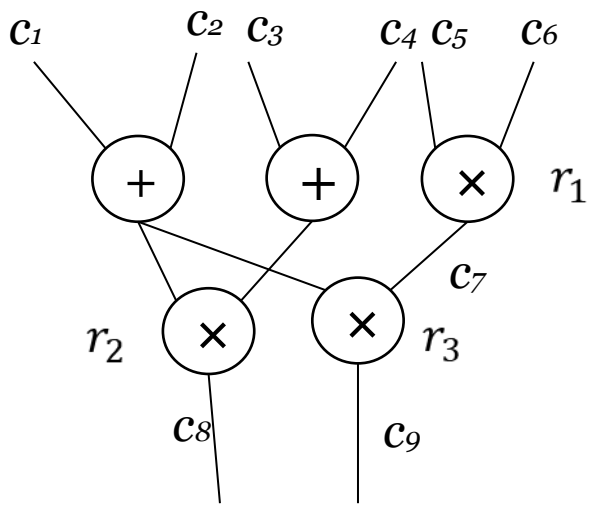


Encoding circuits to polynomials



Circuit SAT to polynomial division

- $p(x) = (\sum_{i=1}^{m} c_i \times v_i(x)) \times (\sum_{i=1}^{m} c_i \times w_i(x)) - (\sum_{i=1}^{m} c_i \times y_i(x))$



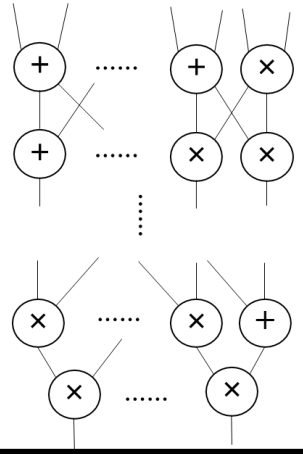
Circuit SAT to polynomial division

- $p(x) = (\sum_{i=1}^{m} c_i \times v_i(x)) \times (\sum_{i=1}^{m} c_i \times w_i(x)) - (\sum_{i=1}^{m} c_i \times y_i(x))$

- Target polynomial: $t(x) = (x - r_1)(x - r_2)(x - r_3)$

Quadratic arithmetic program (QAP)

“Bad” generic verifiable computation



server

client

result = C(data) +

- Compute all polynomials with only the circuit
- On some data, send me all c_i s
- Compute p(x), check it is divisible by t(x)

data

To be continued…

- SNARK from QAP using bilinear map
- Zero knowledge
- Pros and cons