UNIVERSITY OF OREGON

PartialGC: A server-aided system for saving GC state

Benjamin Mood, Debayan Gupta , Kevin Butler, and Joan Feigenbaum

OSIRIS Oregon Systems Infrastructure Research & Information Security Laboratory

February 21, 2014

Data Analysis

Privacy Preserving Data Analysis

Data Analysis

Privacy Preserving Data Analysis

Data Initialization

Data Analysis

UNIVERSITY OF OREGON

Privacy Preserving Data Analysis

Data Initializati

Receive Report

Data Analysis

Privacy Preserving Data Analysis

Data Analysis

UNIVERSITY OF OREGON

Privacy Preserving Data Analysis

Data Initializati	Receive Report	Update Informati	Receive Report

Data Analysis

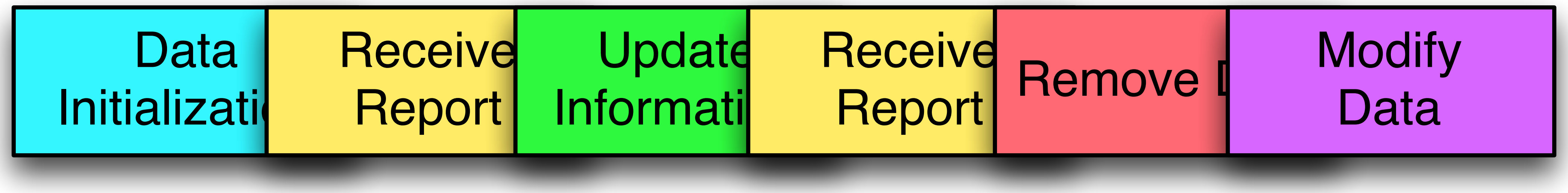
UNIVERSITY OF OREGON

Privacy Preserving Data Analysis

Data Initializati	Receive Report	Update Informati	Receive Report	Remove Data

Data Analysis

Privacy Preserving Data Analysis



Data Analysis

UNIVERSITY OF OREGON

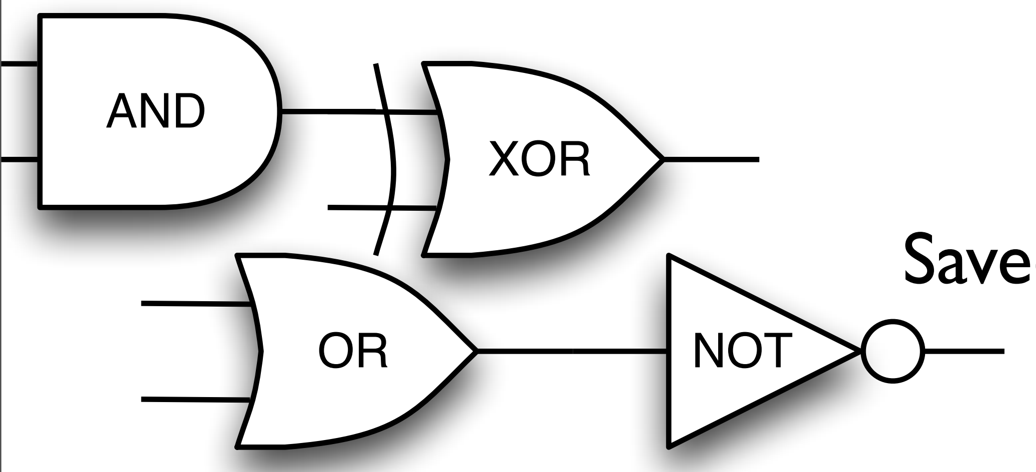
Privacy Preserving Data Analysis

Data Initializati	Receive Report	Update Informati	Receive Report	Remove D	Modify Data	Receive Report

Overall Idea:

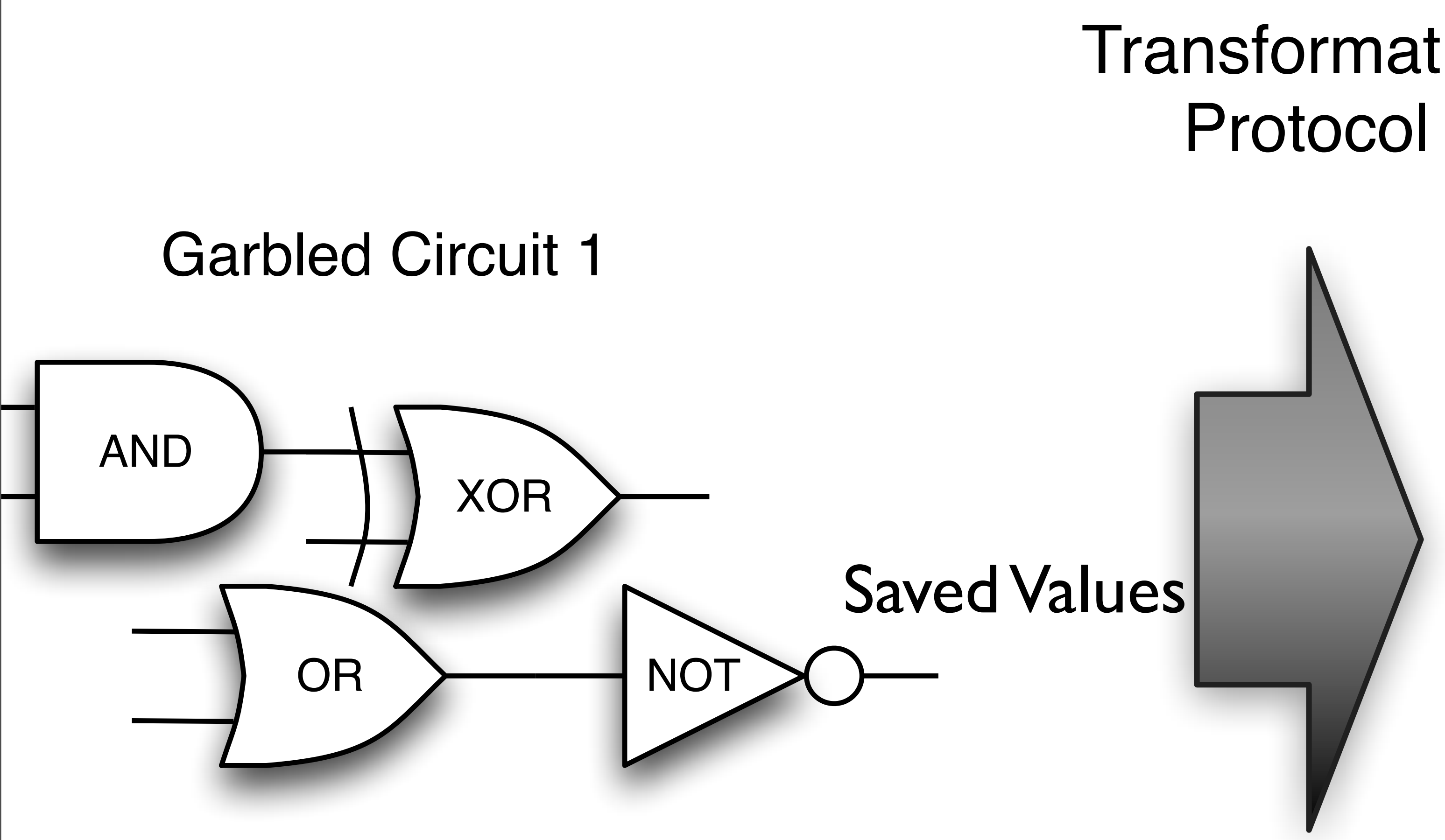
Overall Idea:

Garbled Circuit 1

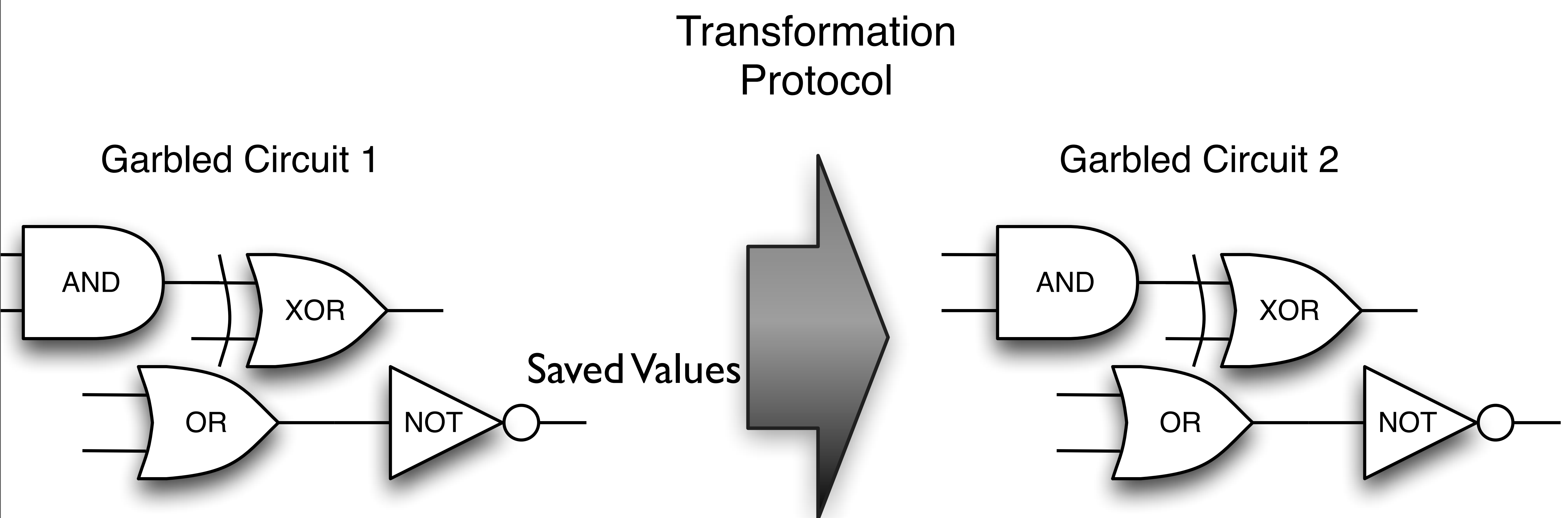


Saved Values

Overall Idea:



Overall Idea:

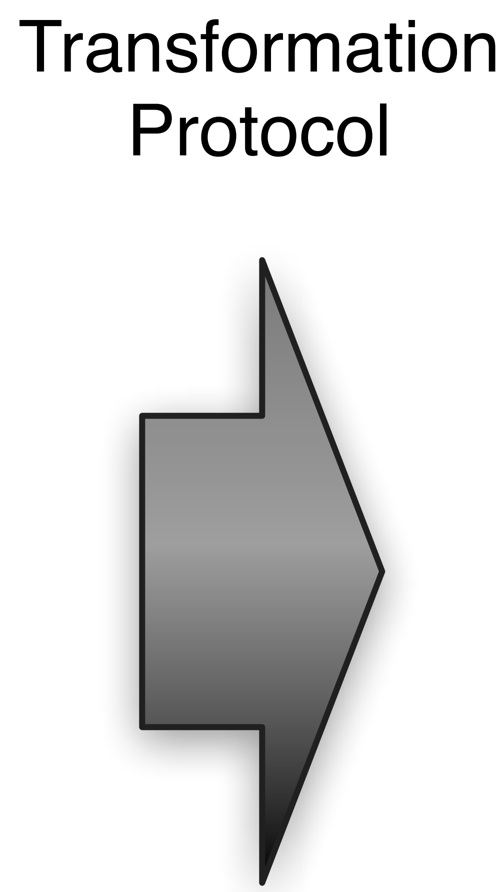


Outline

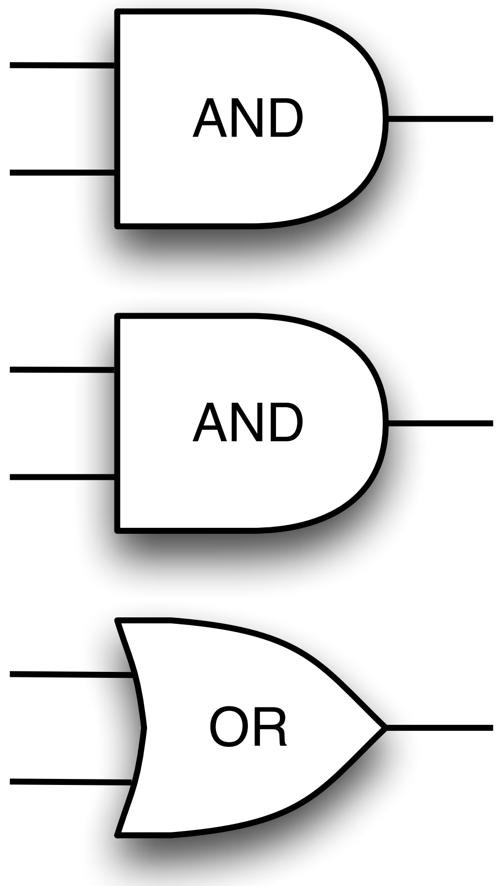
- Transformation
- Checking Transformation
- Server Aided Protocol
- Results

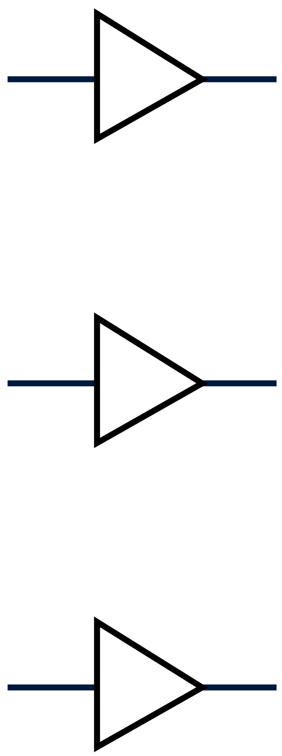
Transforming Wires

- Generator creates garbled gates that transform the wires that work in one garbled circuit to wires that work in another garbled circuit.



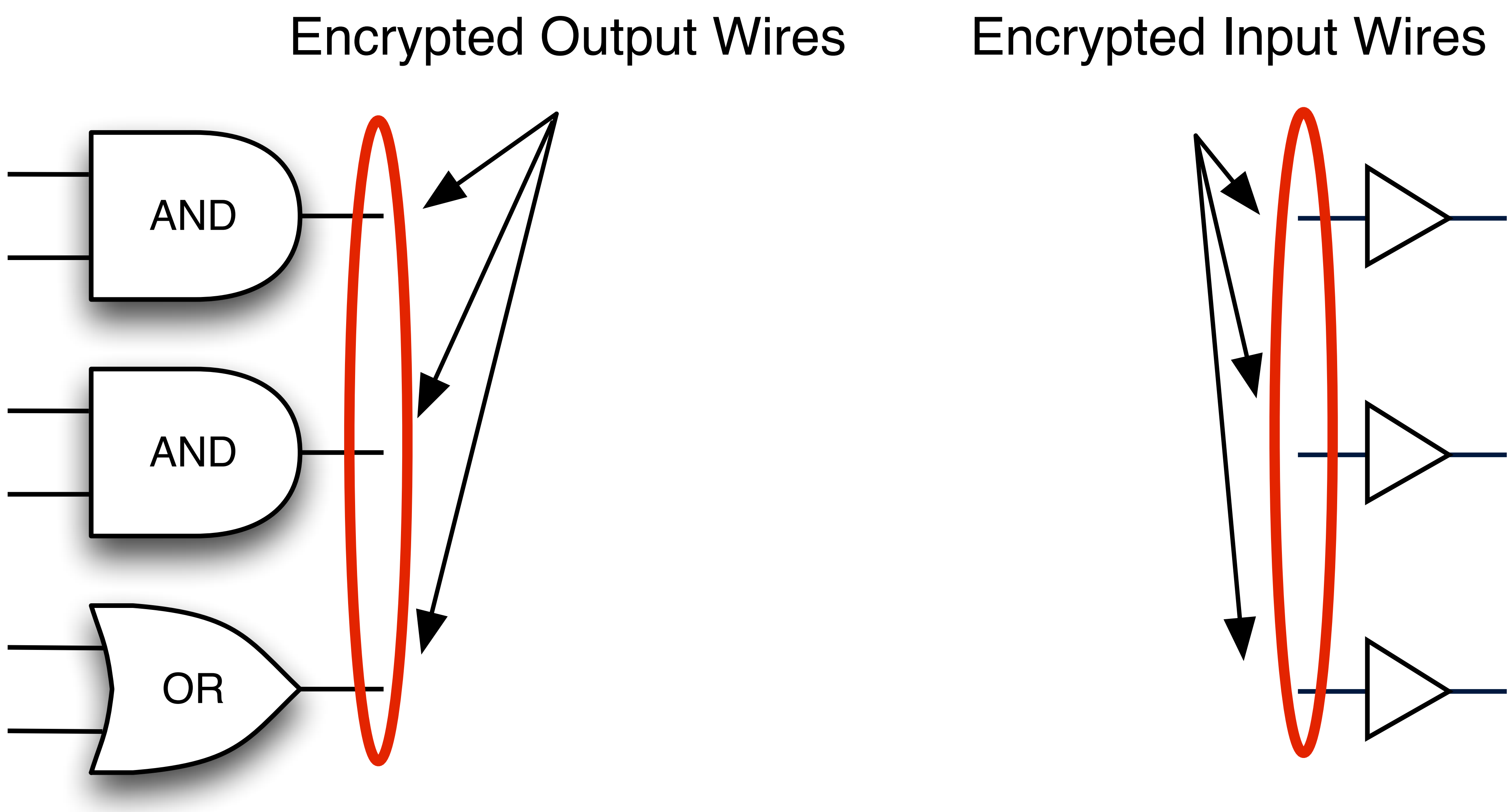
Transformation Details





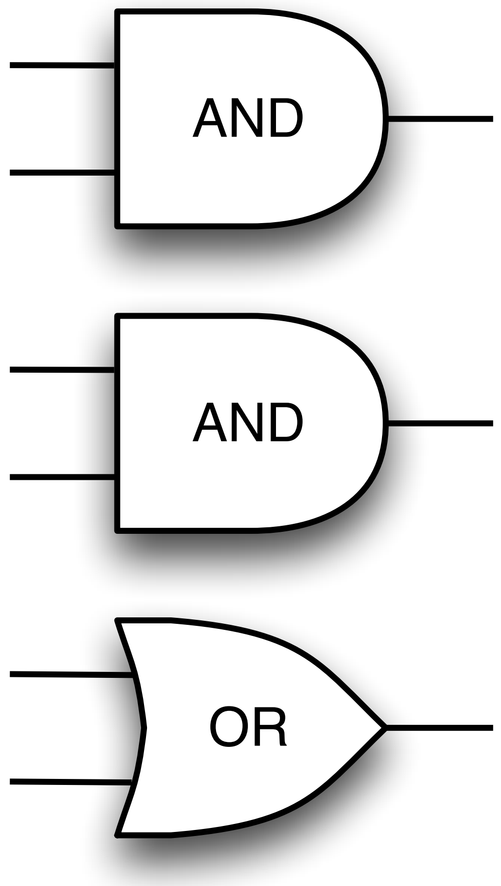
* = once per circuit

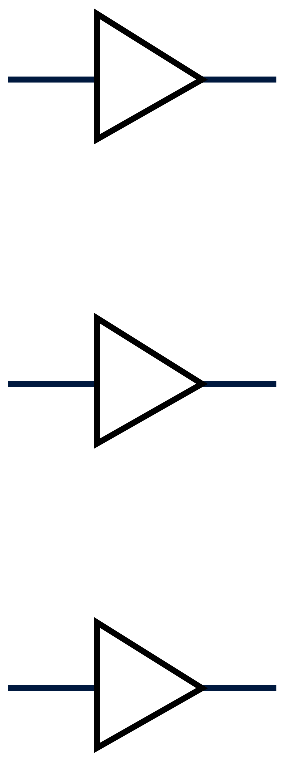
Transformation Details



* = once per circuit

Transformation Details



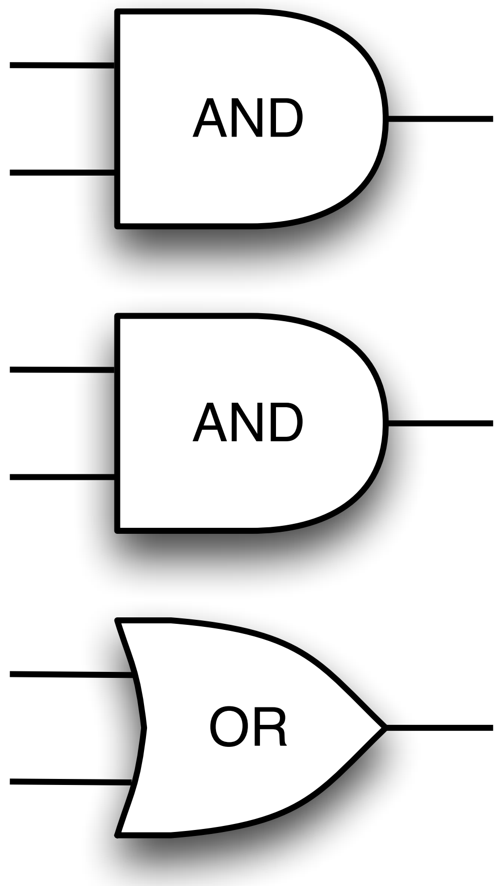


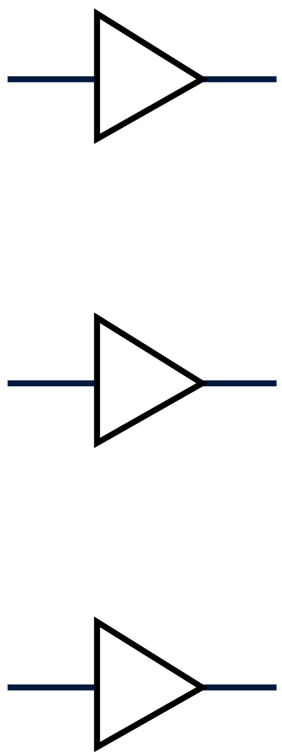
* = once per circuit

Transformation Details

UNIVERSITY OF OREGON

Generator

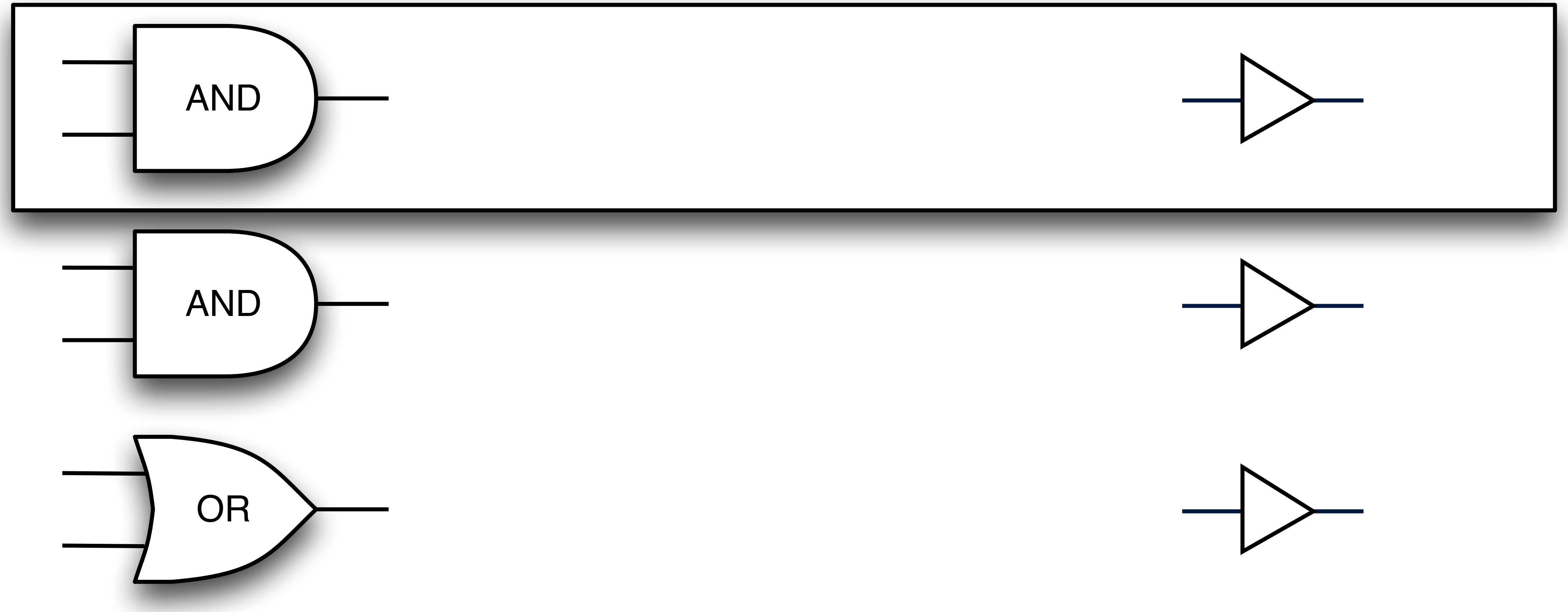




* = once per circuit

Transformation Details

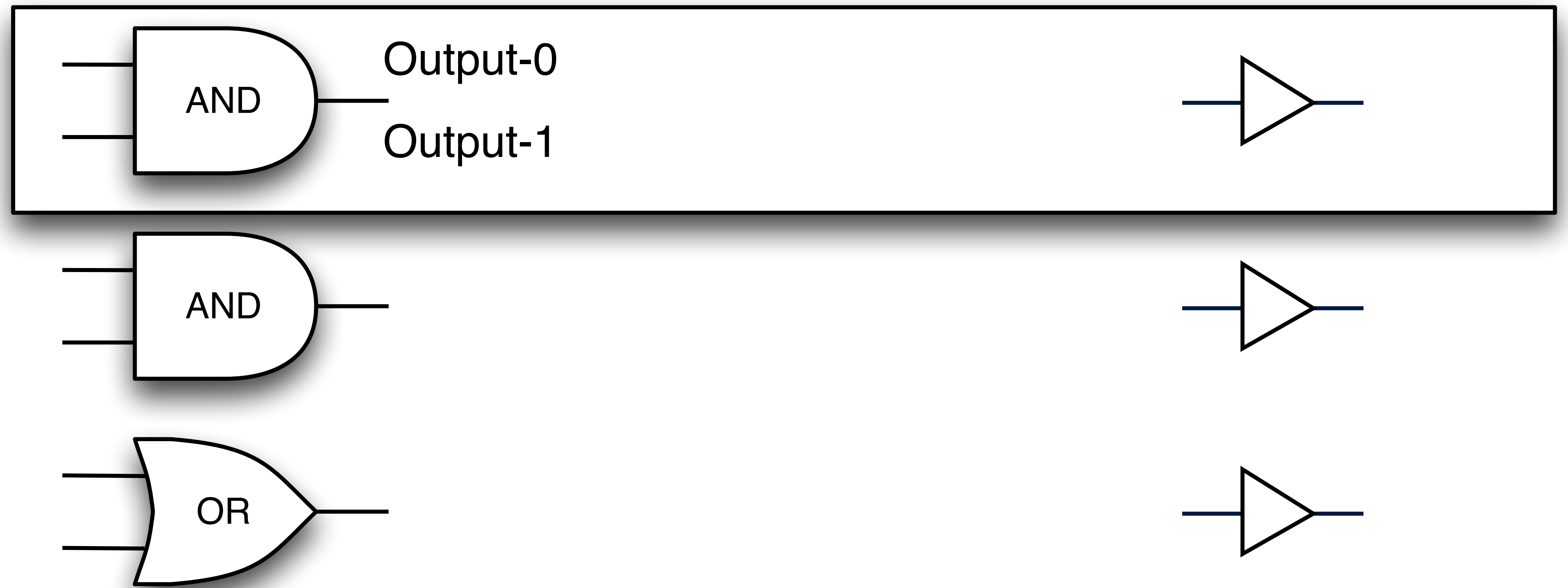
Generator



* = once per circuit

Transformation Details

Generator

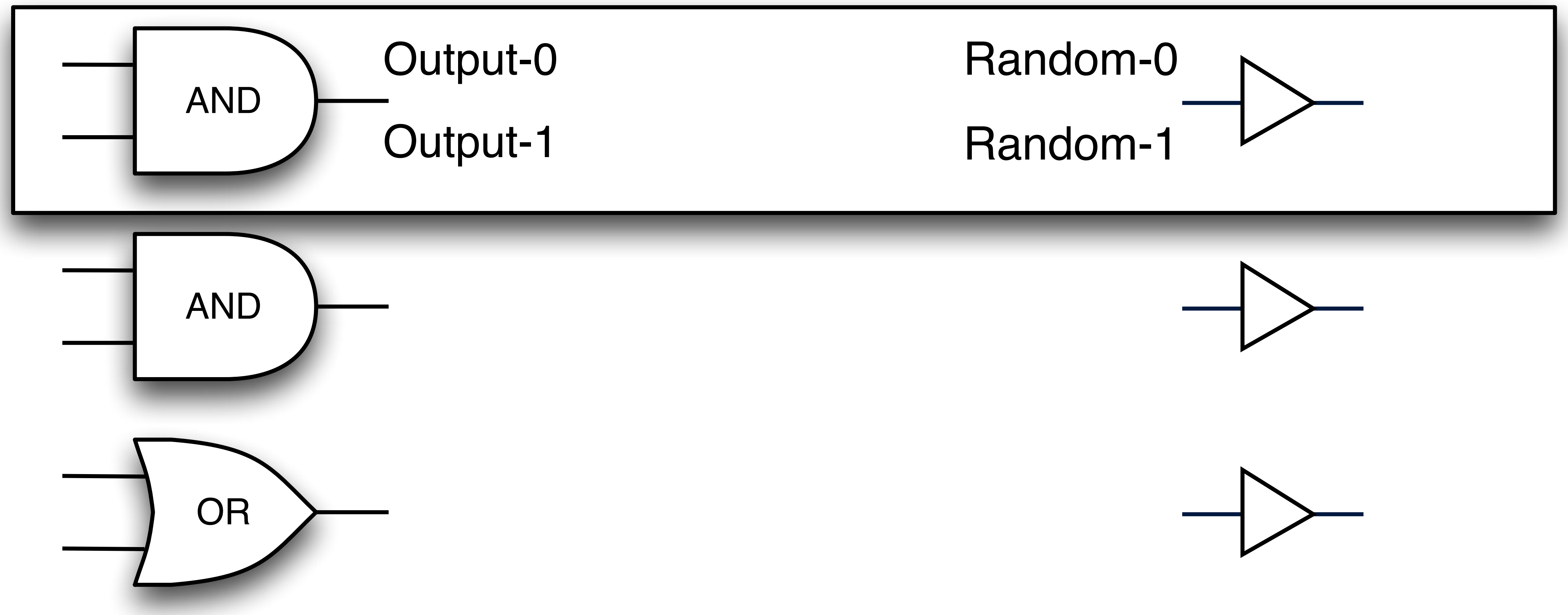


* = once per circuit

Transformation Details

UNIVERSITY OF OREGON

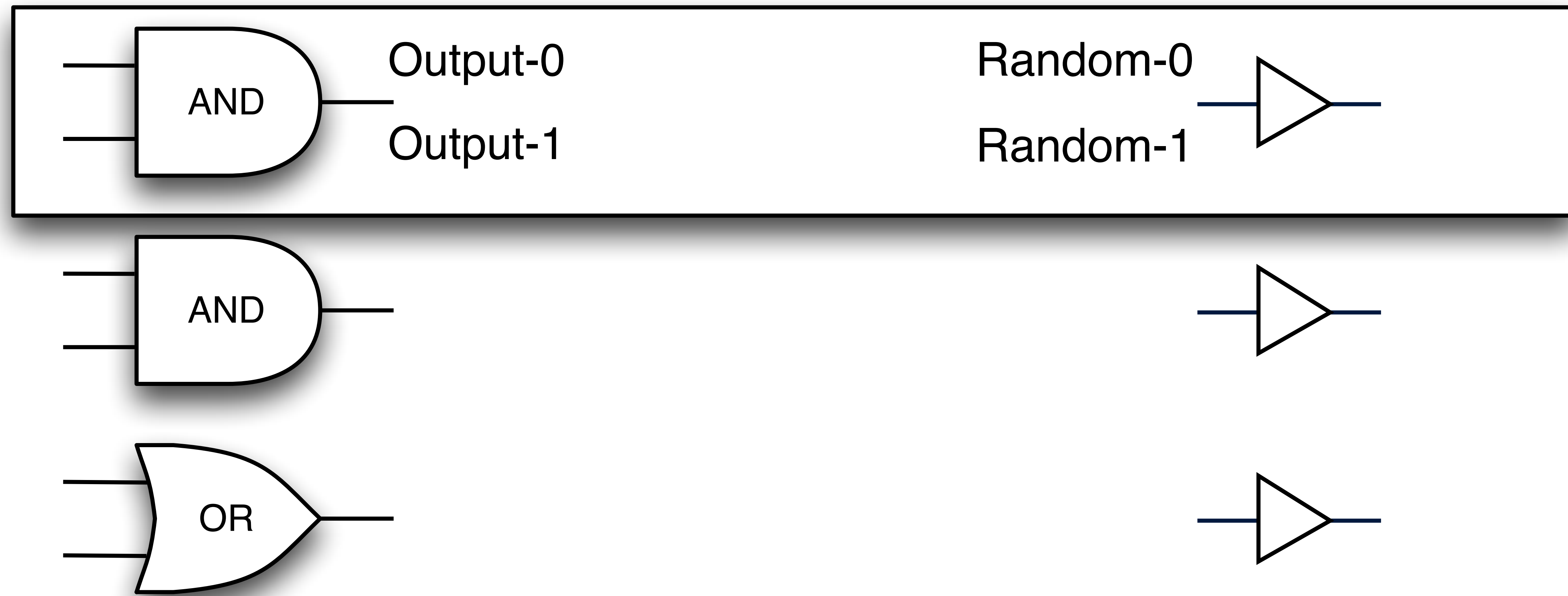
Generator



* = once per circuit

Transformation Details

Generator



nonce = PRNG.rand() *

Transform-0 = hash(Output-0 $\oplus$ nonce) $\oplus$ Random-0

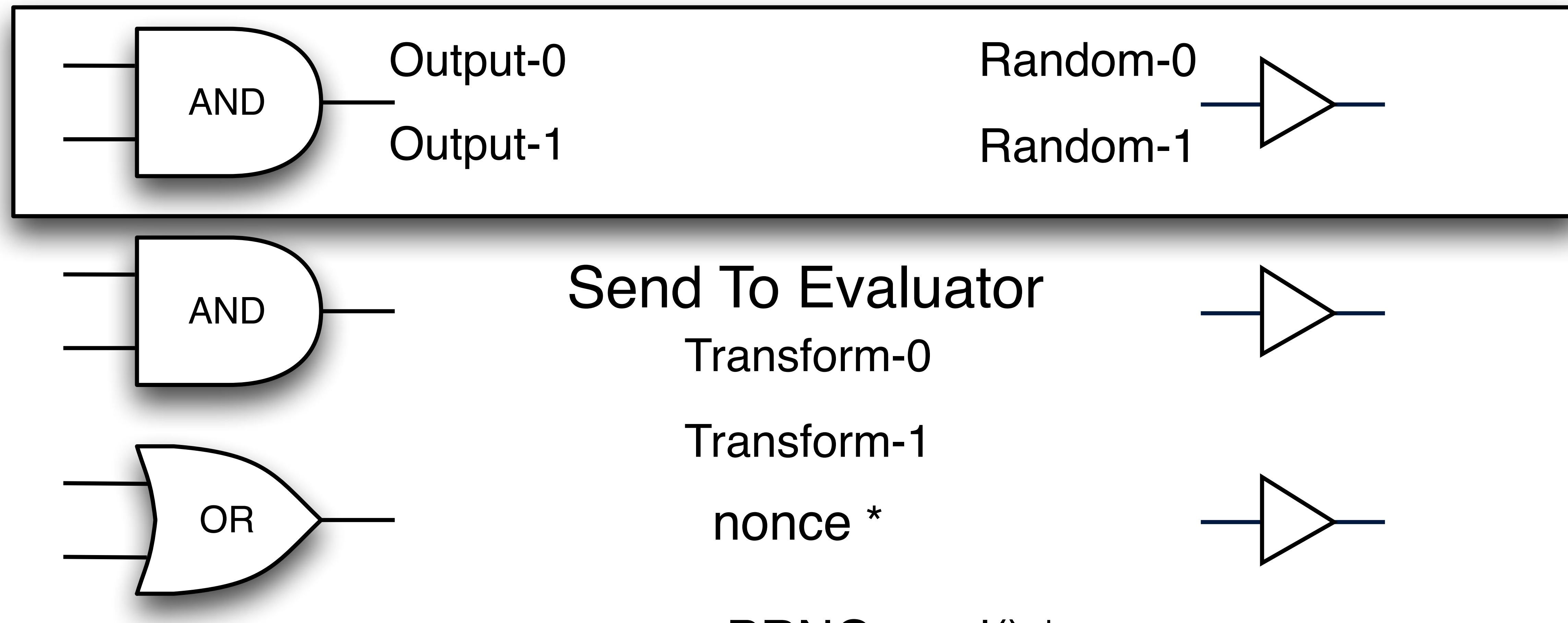
Transform-1 = hash(Output-1 $\oplus$ nonce) $\oplus$ Random-1

* = once per circuit

Transformation Details

UNIVERSITY OF OREGON

Generator



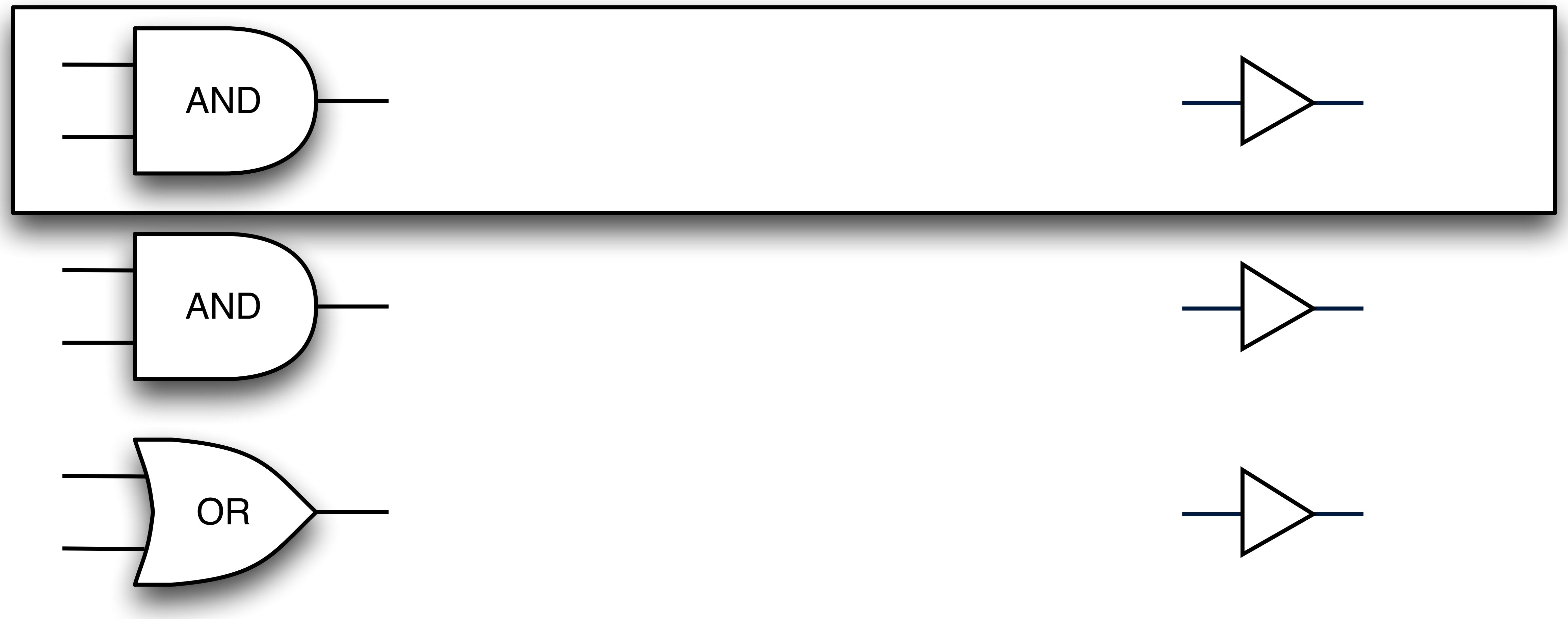
nonce = PRNG.rand() *

$$\text{Transform-0} = \text{hash}(\text{Output-0} \oplus \text{nonce}) \oplus \text{Random-0}$$

$$\text{Transform-1} = \text{hash}(\text{Output-1} \oplus \text{nonce}) \oplus \text{Random-1}$$

* = once per circuit

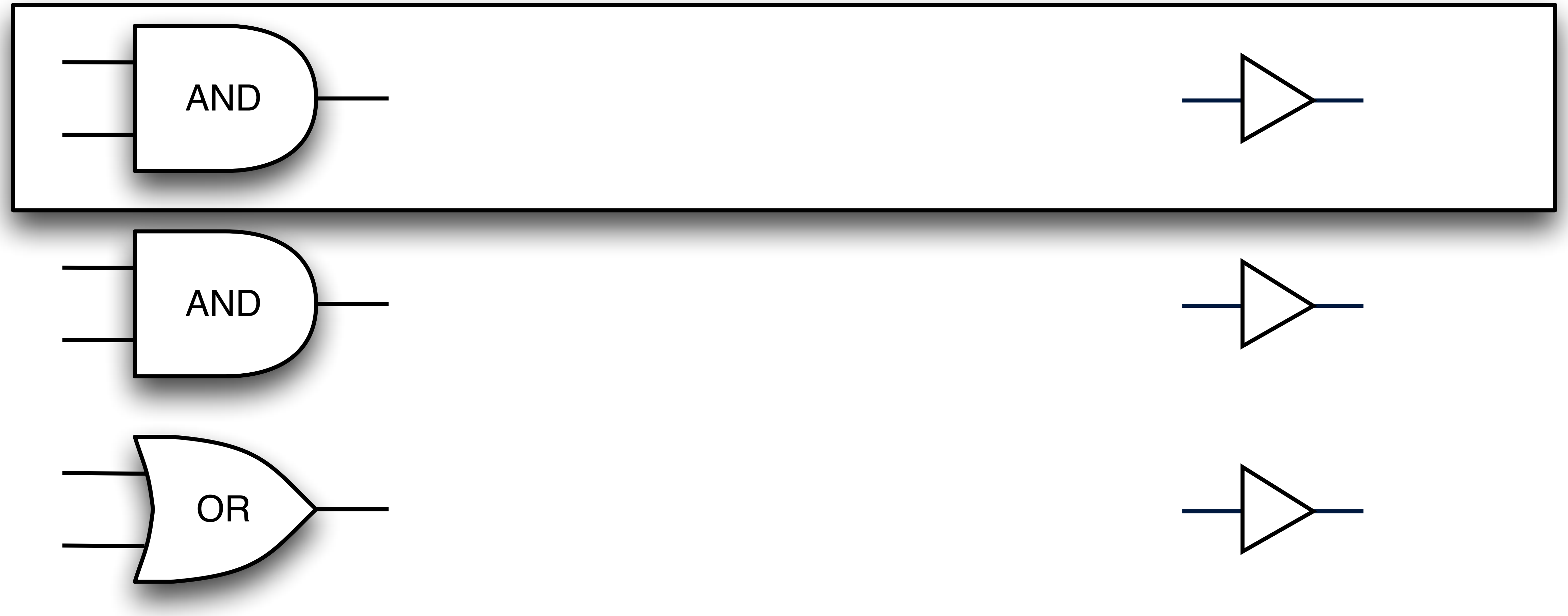
Transformation Details



* = once per circuit

Transformation Details

Evaluator

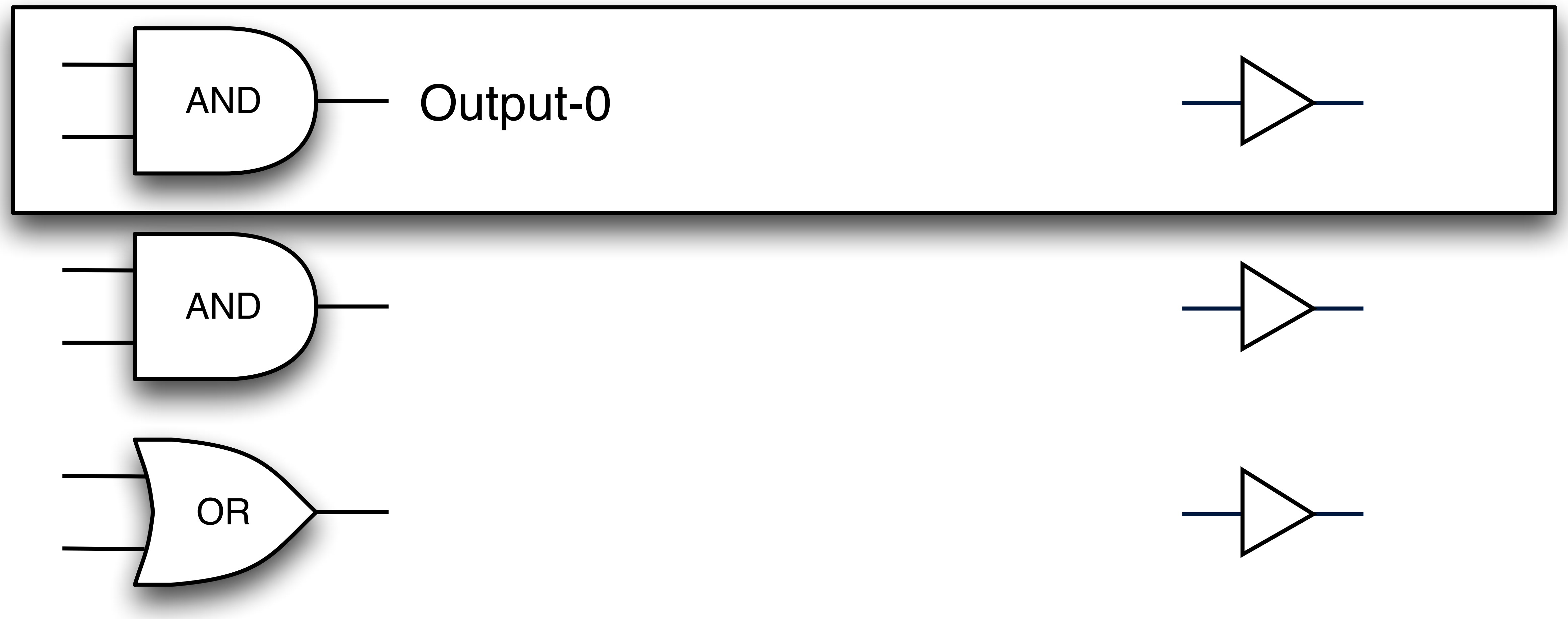


* = once per circuit

Transformation Details

UNIVERSITY OF OREGON

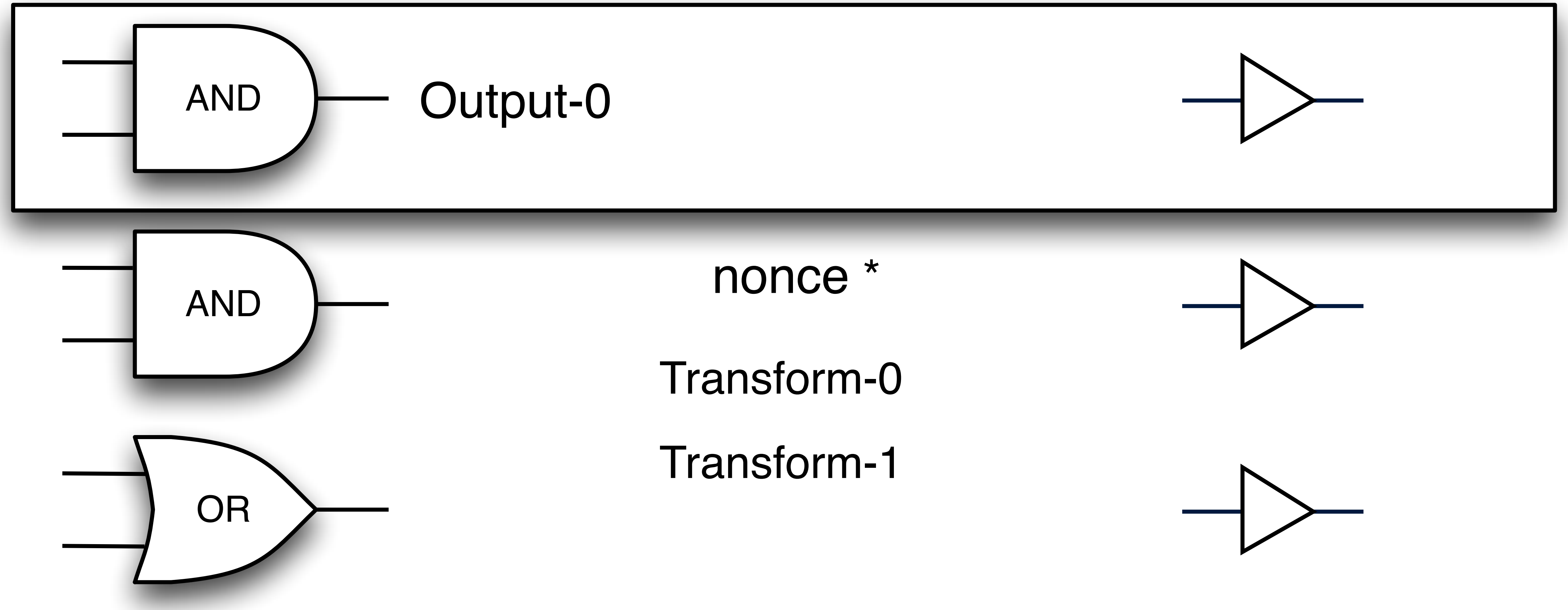
Evaluator



* = once per circuit

Transformation Details

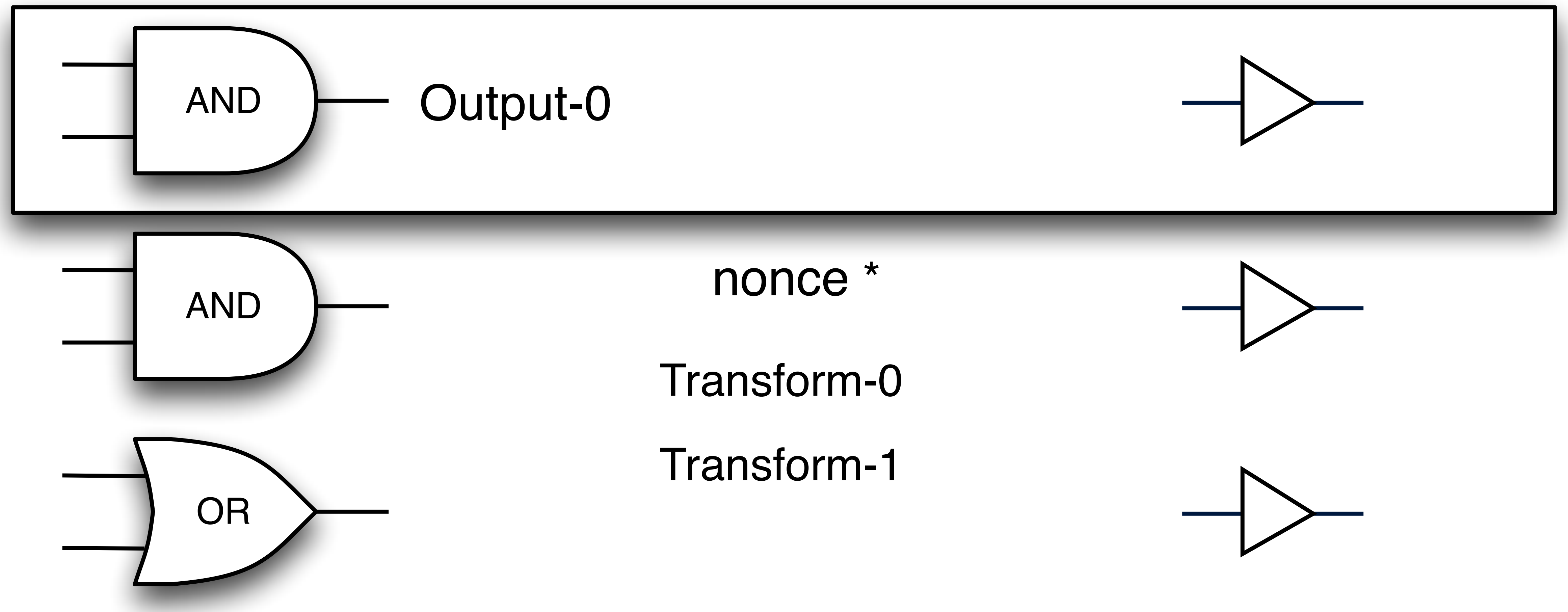
Evaluator



* = once per circuit

Transformation Details

Evaluator



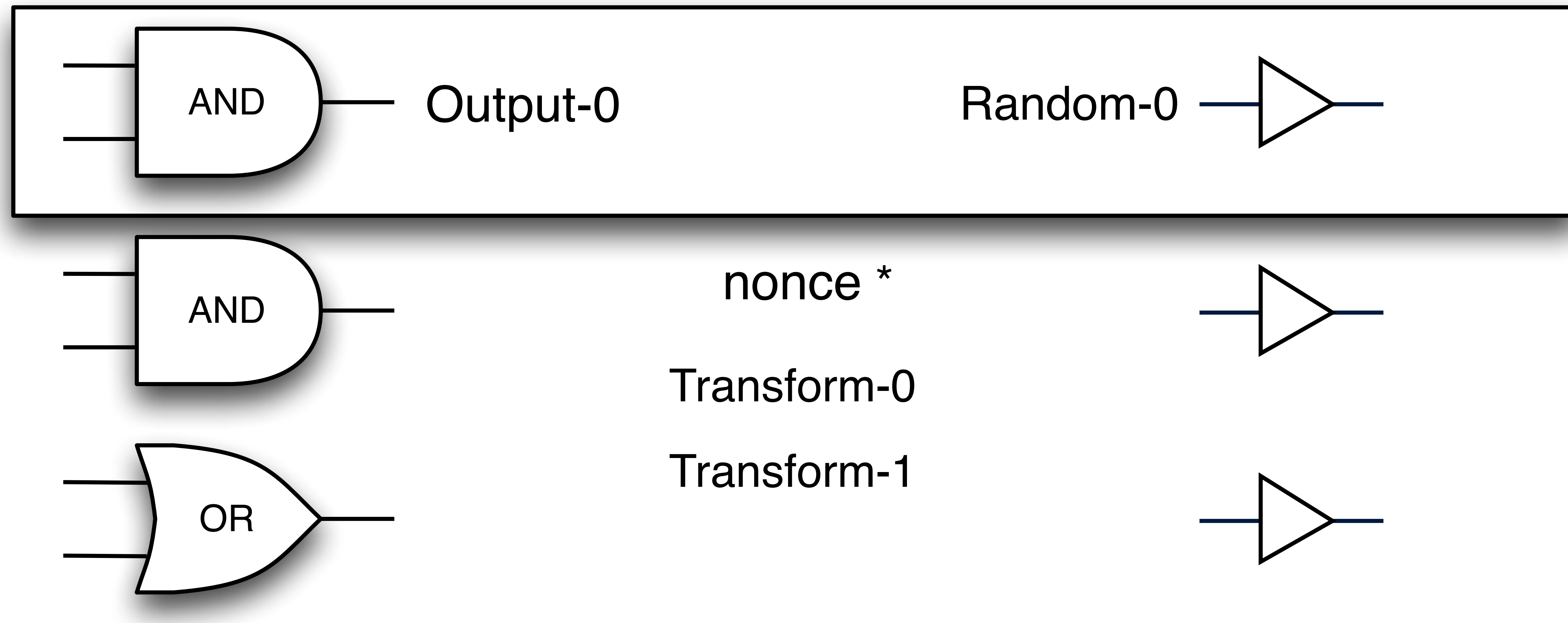
Random-0 = hash(Output-0 $\oplus$ nonce) $\oplus$ Transform-0

* = once per circuit

Transformation Details

UNIVERSITY OF OREGON

Evaluator



Random-0 = hash(Output-0 $\oplus$ nonce) $\oplus$ Transform-0

* = once per circuit

How to check?

- Evaluator can save the possible out values for a check circuit and upon receiving the next iteration of that check circuit can verify the transformation is correct.

How to check? cont.

- **Problem:**
  In our base protocol both parties know the check and evaluation split allowing the generator to only disrupt evaluation gates unless we commit.
- If we commit ahead of time we introduce other problems of longevity of the values.

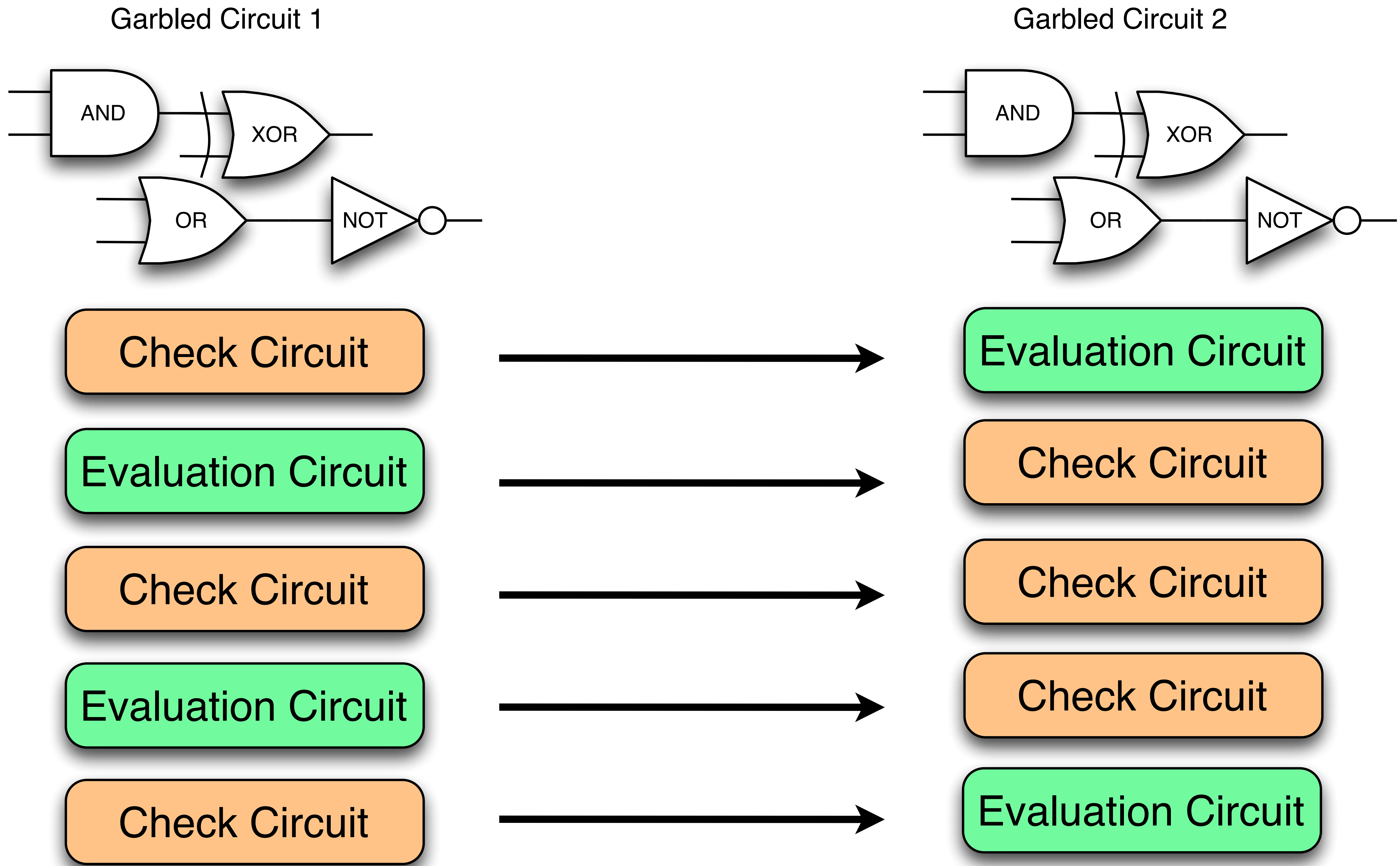
Insight

UNIVERSITY OF OREGON

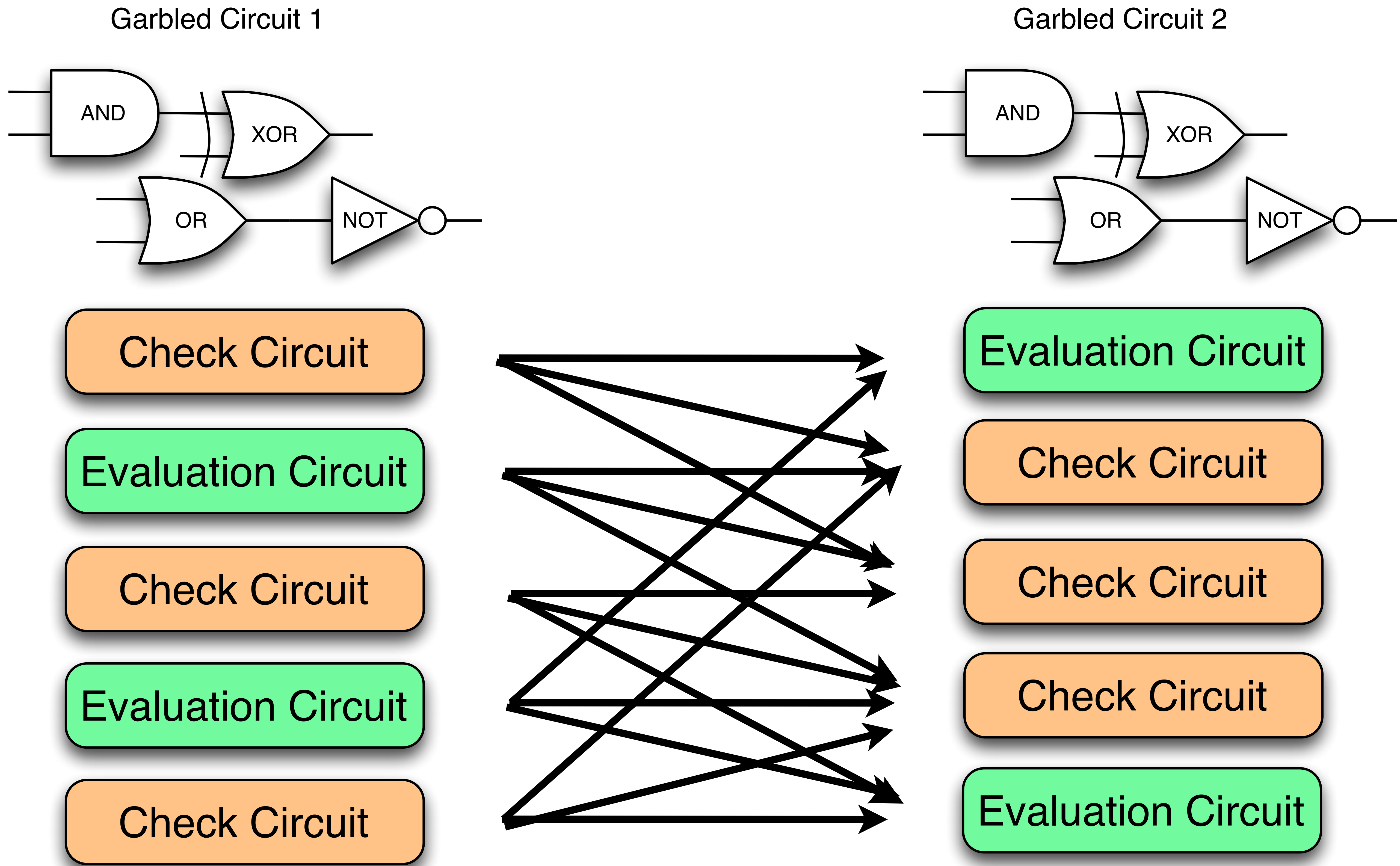
- If the generator does not know the evaluation circuits from the check circuits, then he has to send correct transformation gates for all circuits.
- This also means the generator, for the entirety of the computation, can never learn the split.

Multiple Cut and Choose

UNIVERSITY OF OREGON

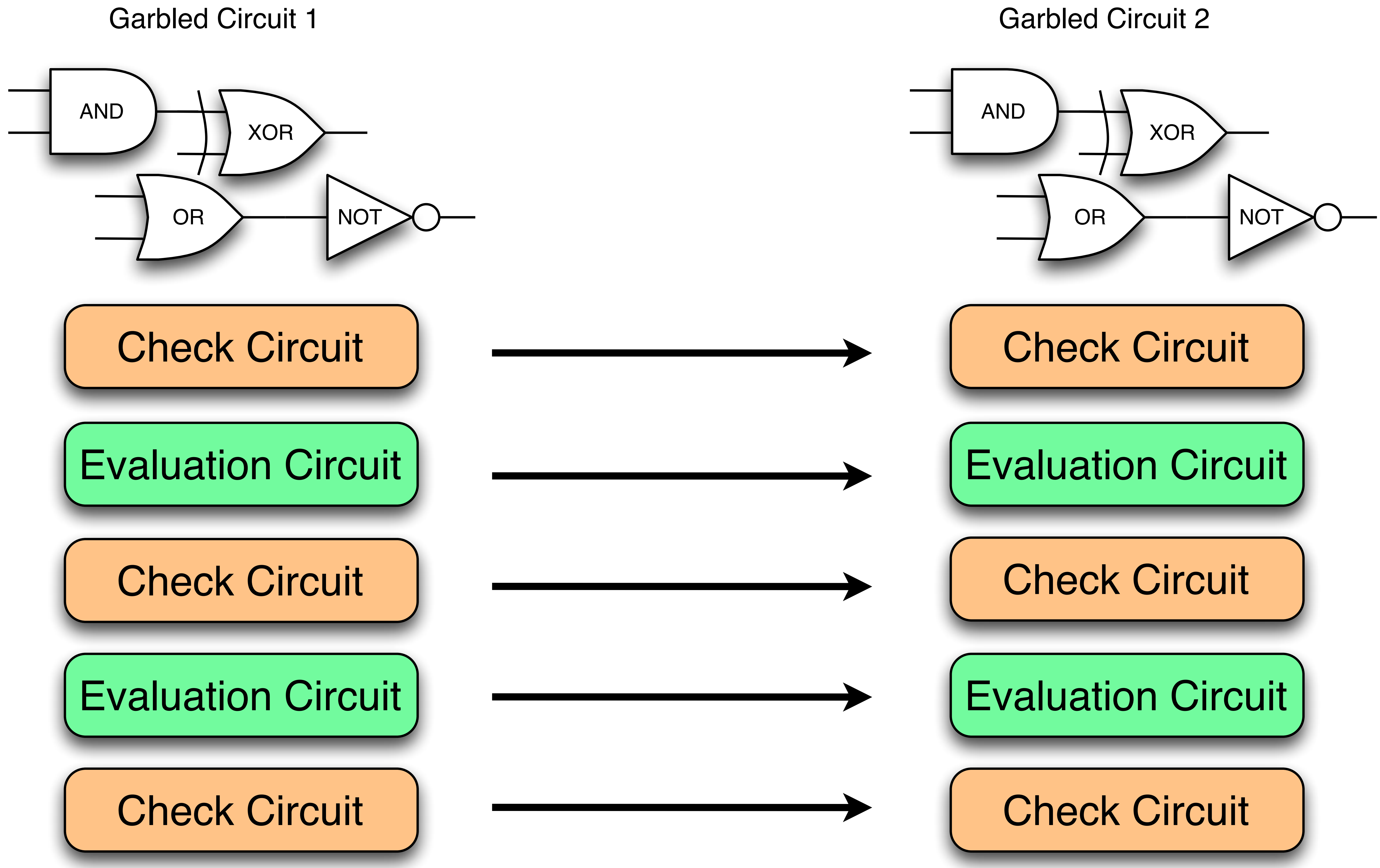


Multiple Cut and Choose



Single Cut and Choose

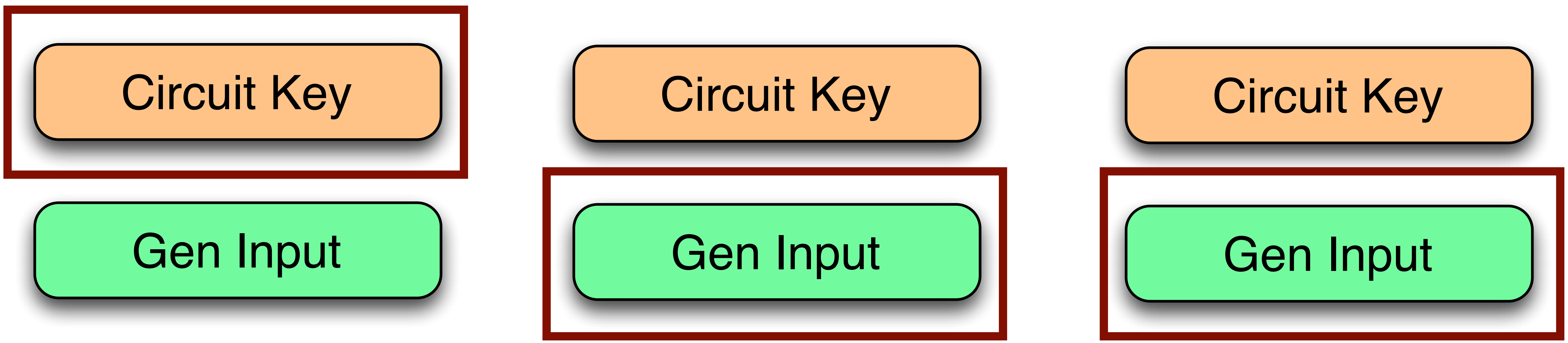
UNIVERSITY OF OREGON



Details

- For the cut and choose, we use OT to select encryption keys as implemented in [SS13].
- In the first computation perform the cut and choose.
- In any subsequent computation use the encryption keys to generate new encryption keys.

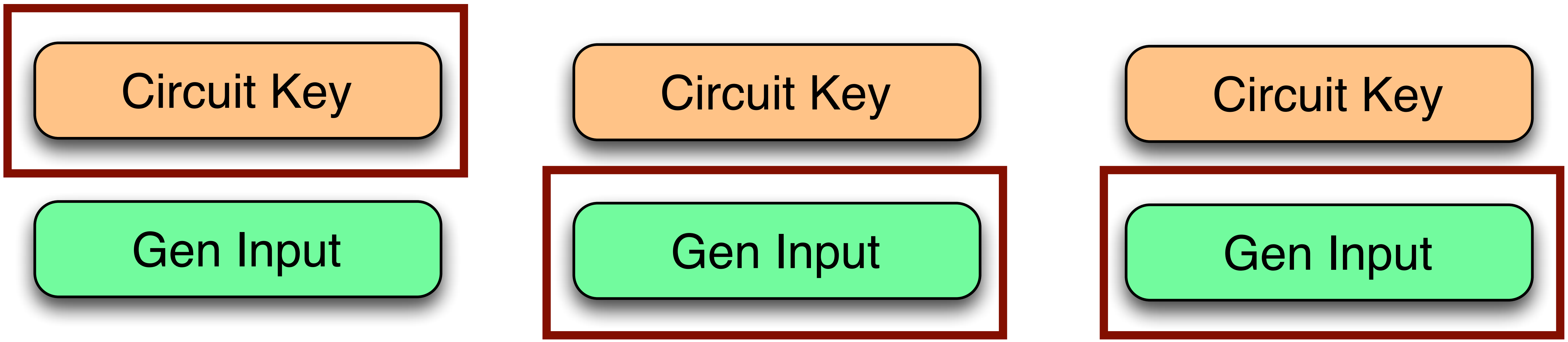
3 Circuit Example



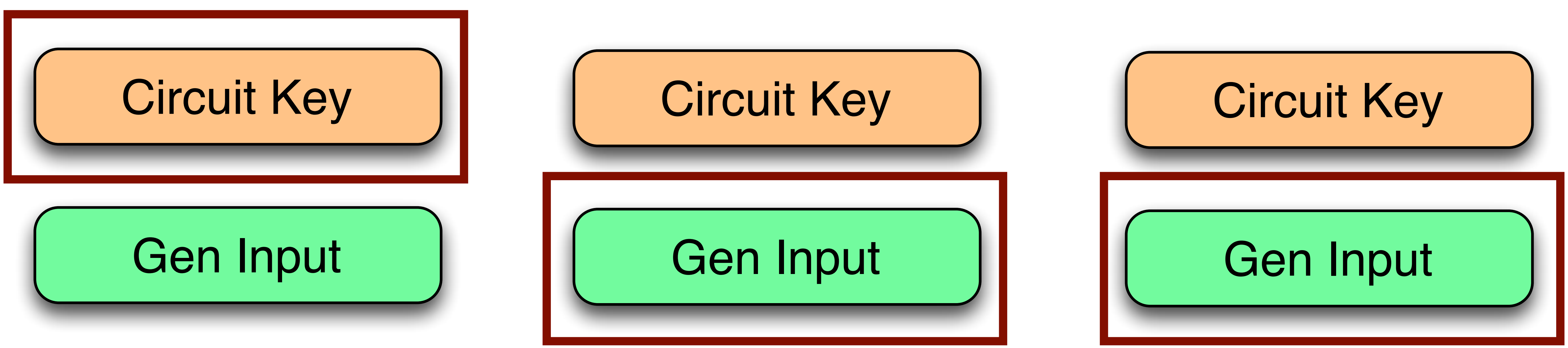
Cut and Choose via OT

3 Circuit Example

UNIVERSITY OF OREGON



Cut and Choose via OT



Checking Transformations

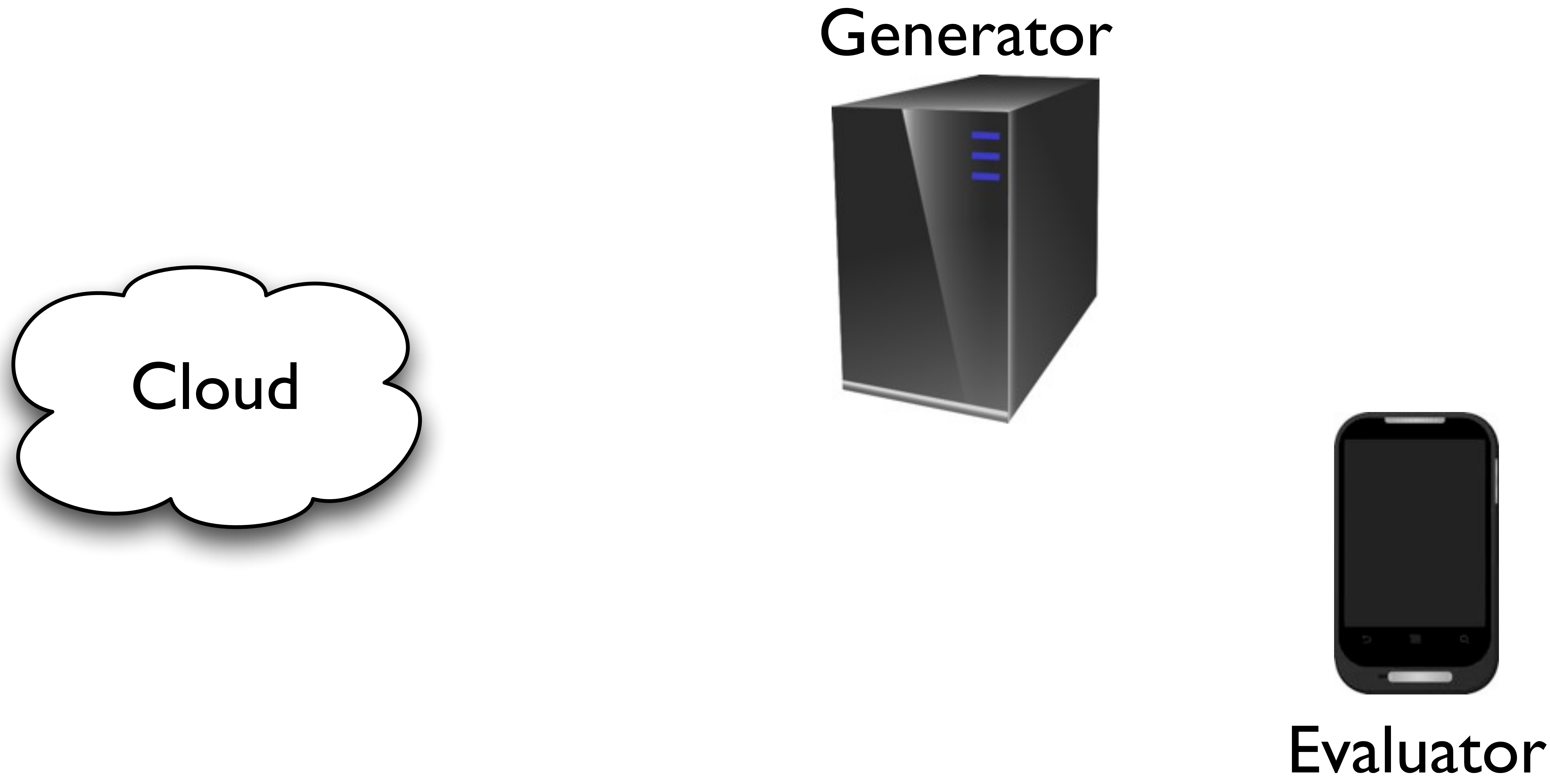
UNIVERSITY OF OREGON

- Generator never learns the check/evaluation circuits
- Evaluator can check how the generator transforms values from one garbled circuit computation to another garbled circuit computation.

Implementation

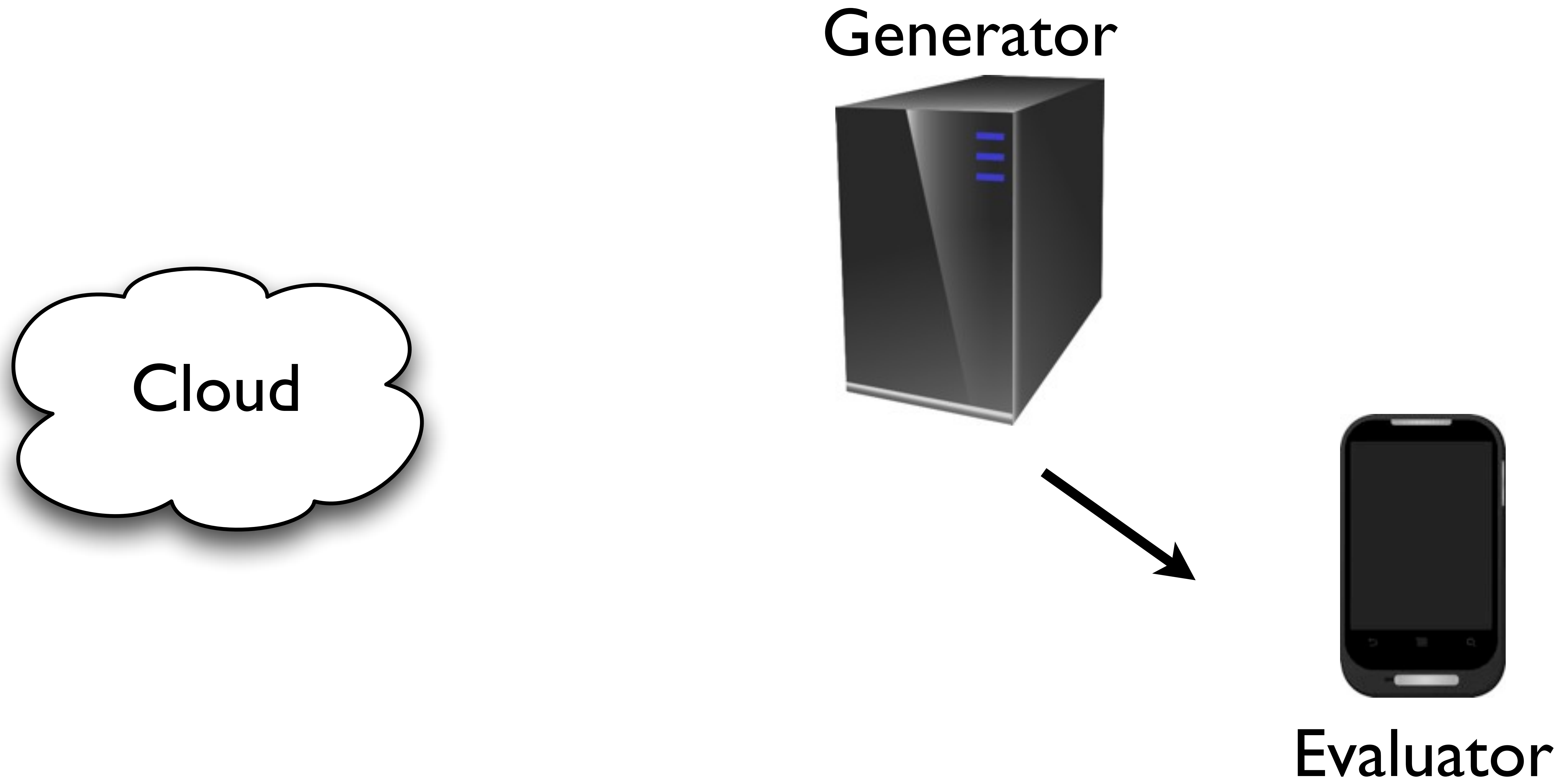
- Server-aided setting
- [CMTB13] system: Outsources the evaluation of a garbled circuit from a mobile device to a high performance server (cloud) with security guarantees.
- Based on [KSS12]

CMTB



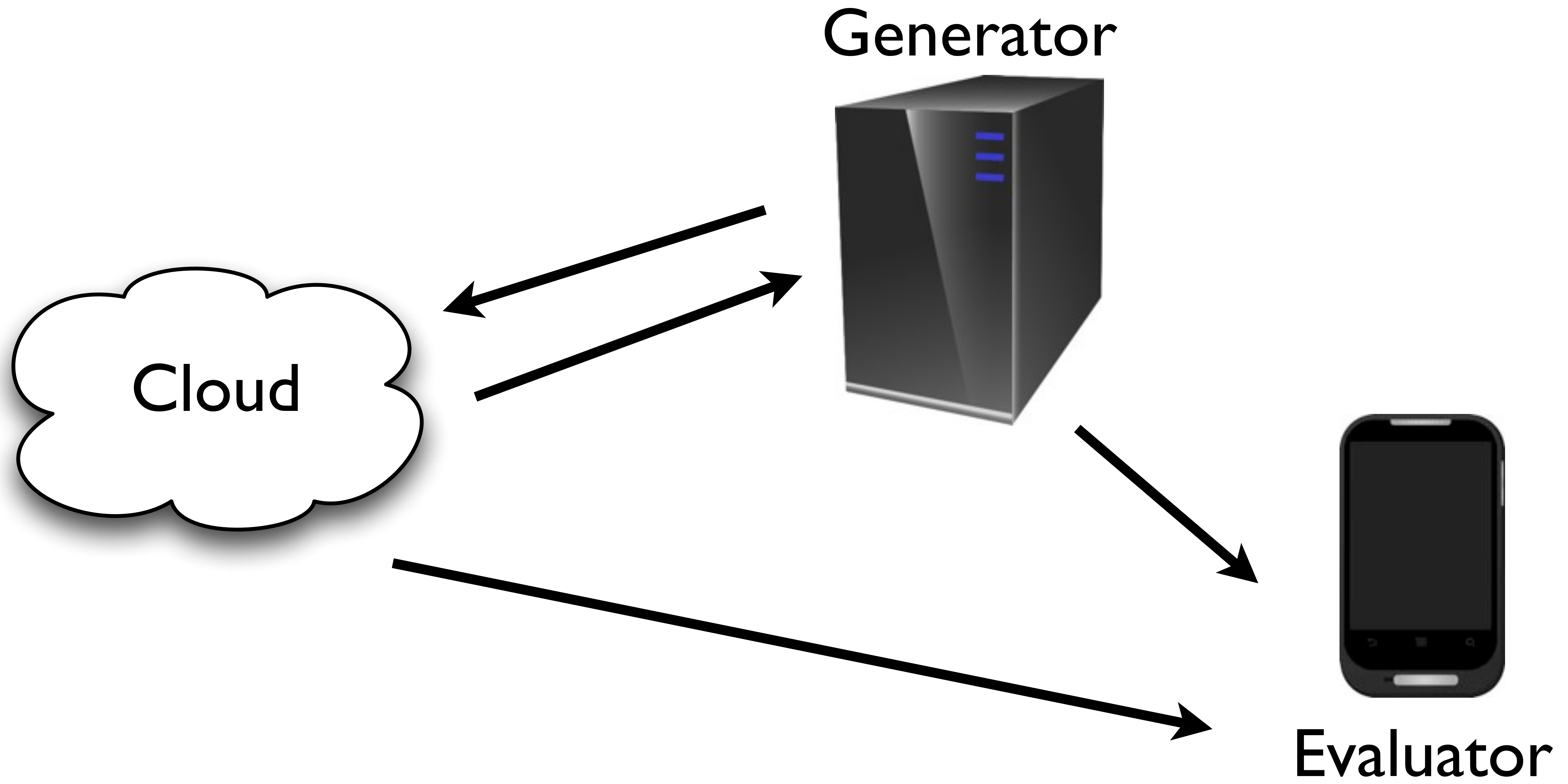
CMTB

UNIVERSITY OF OREGON



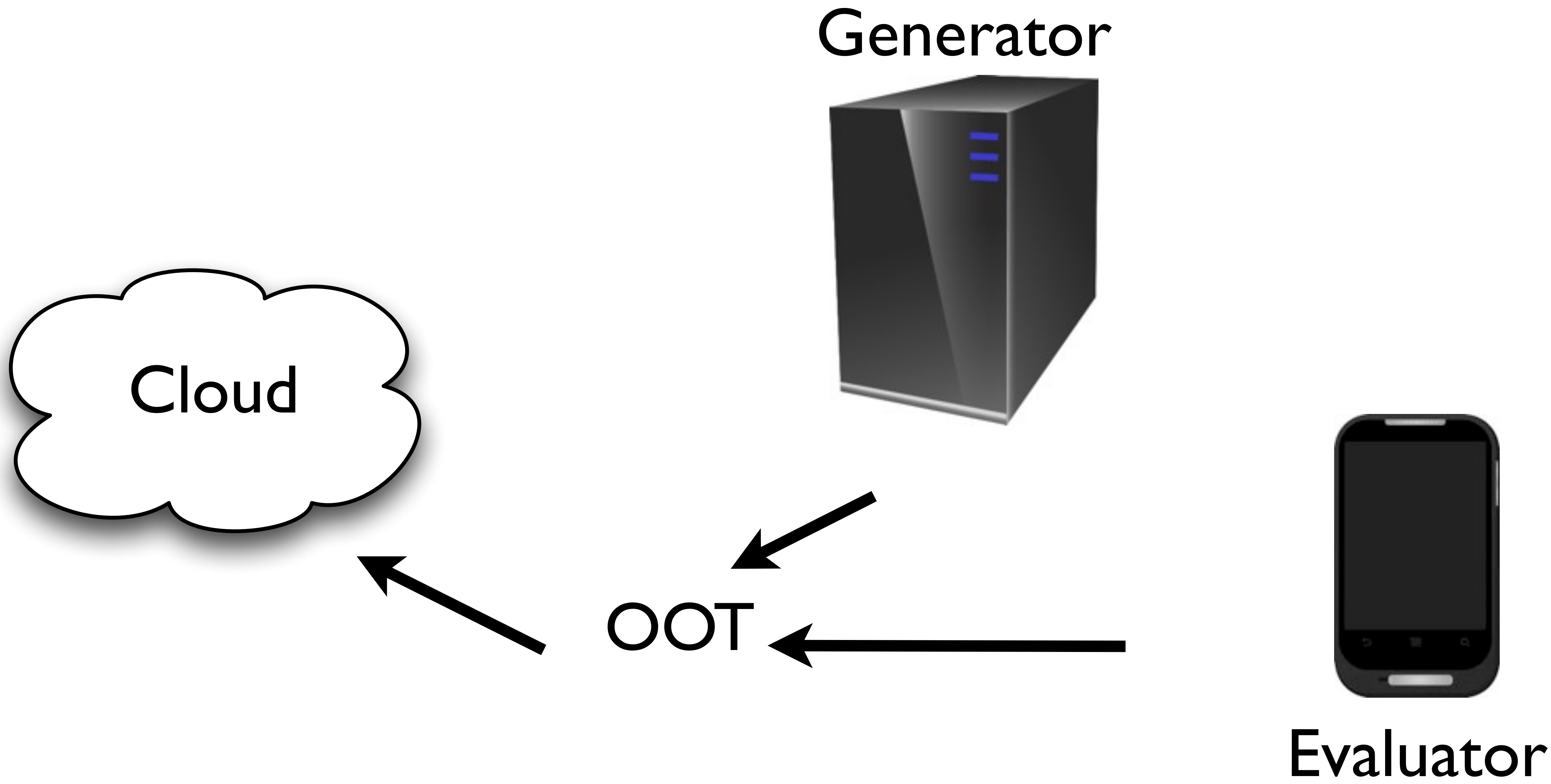
Circuit Commit

CMTB



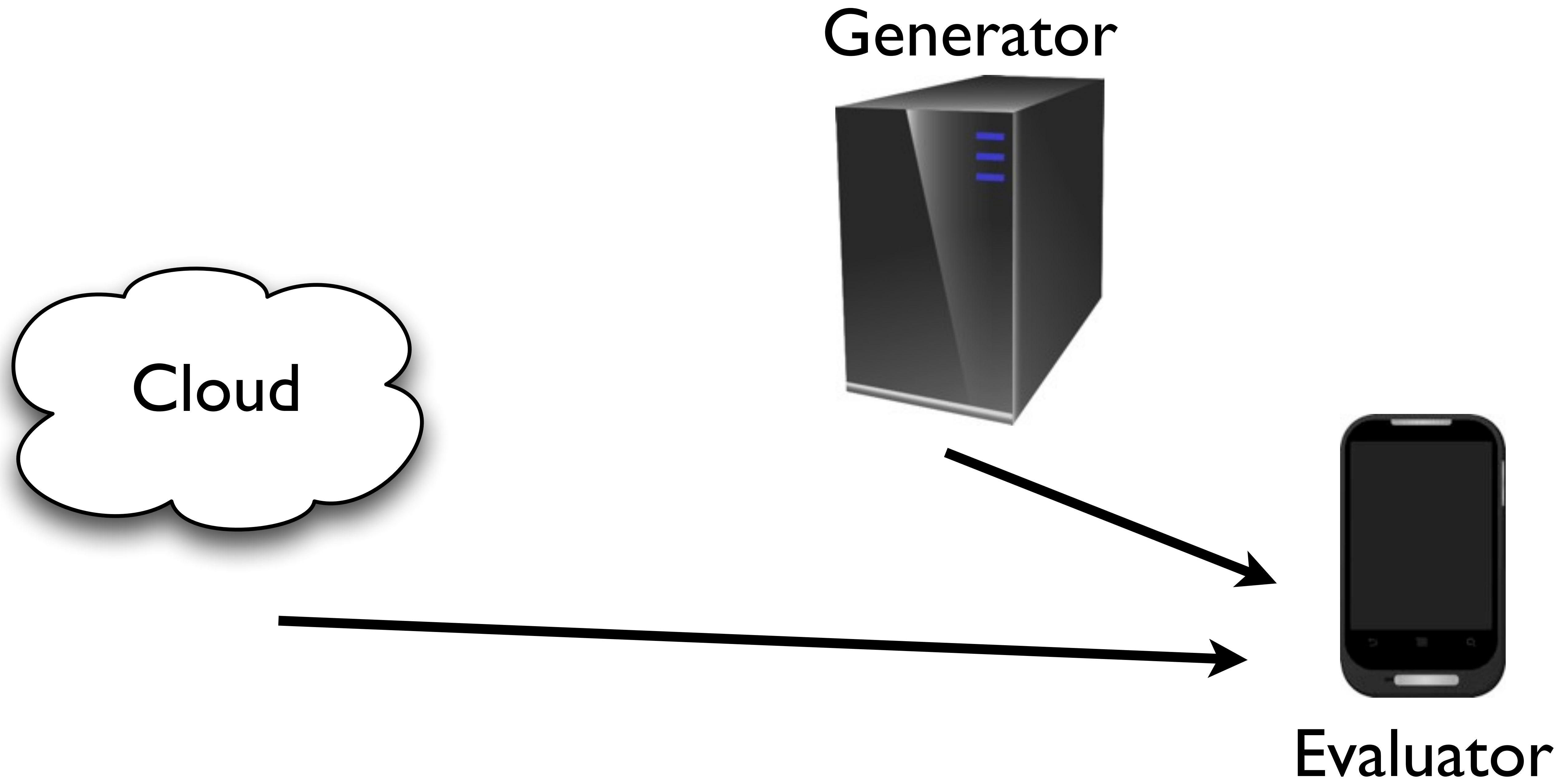
Cut and Choose

CMTB



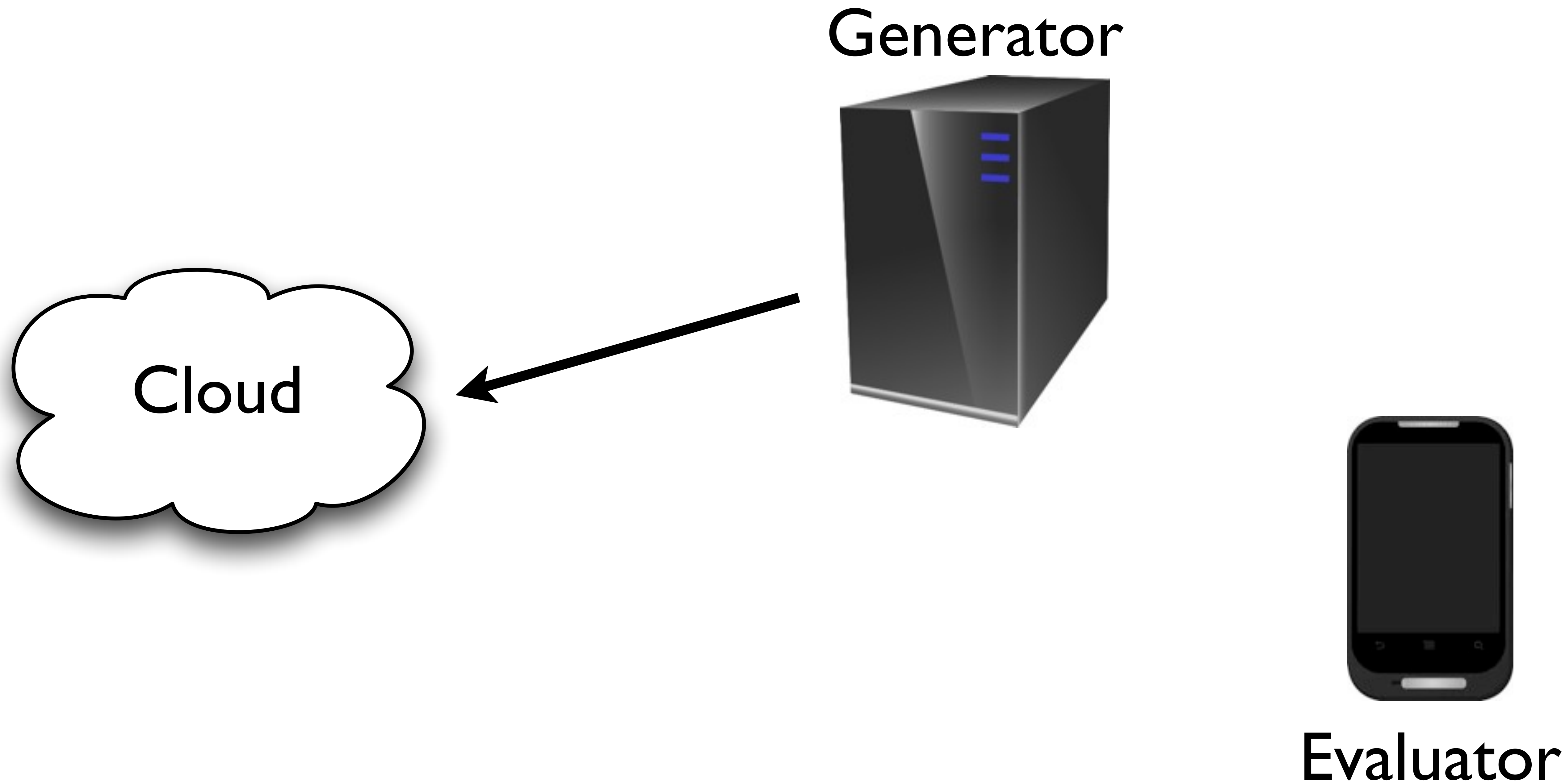
Outsourced Oblivious transfer

CMTB



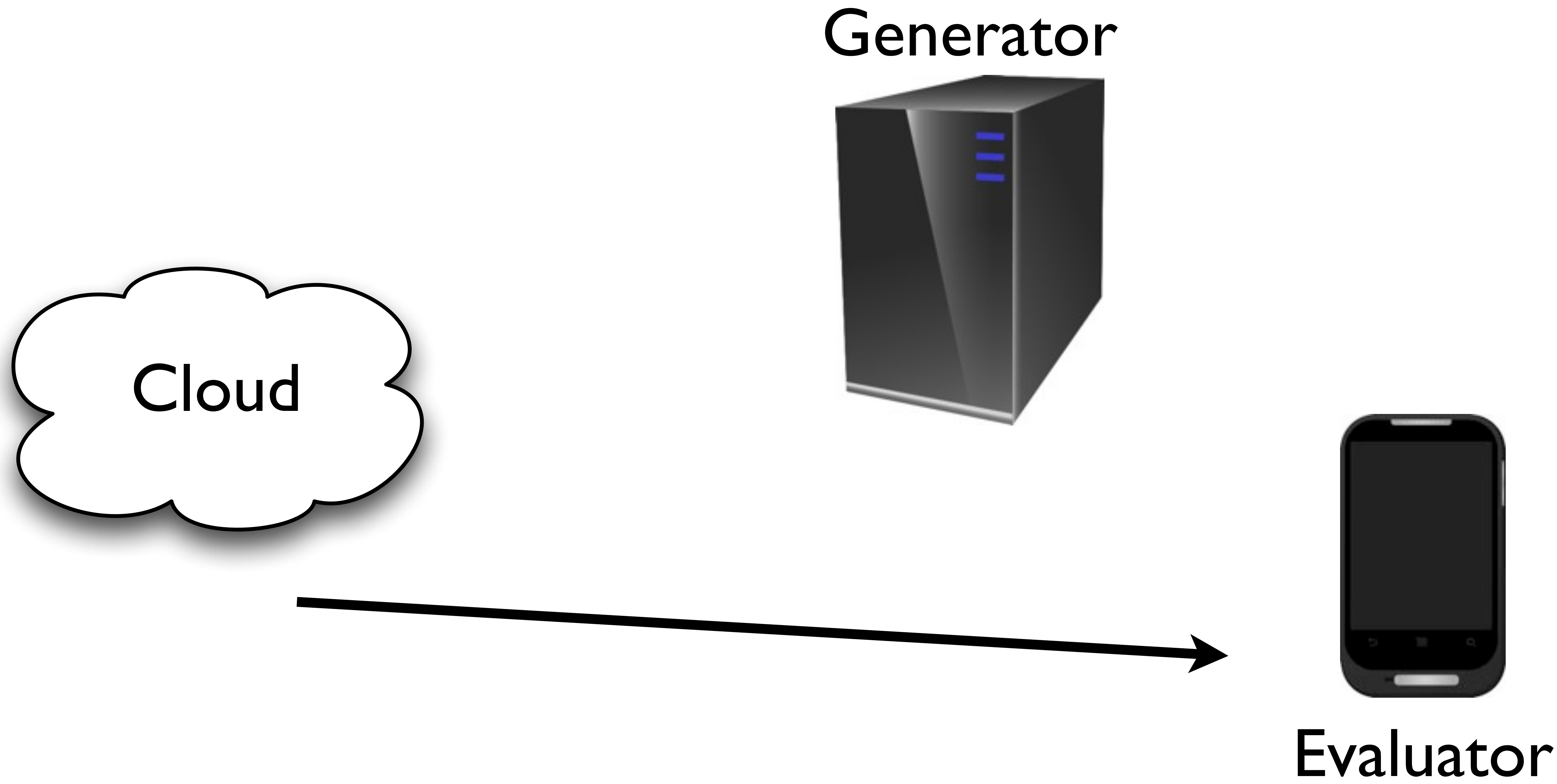
Generator's Input
Consistency Check

CMTB



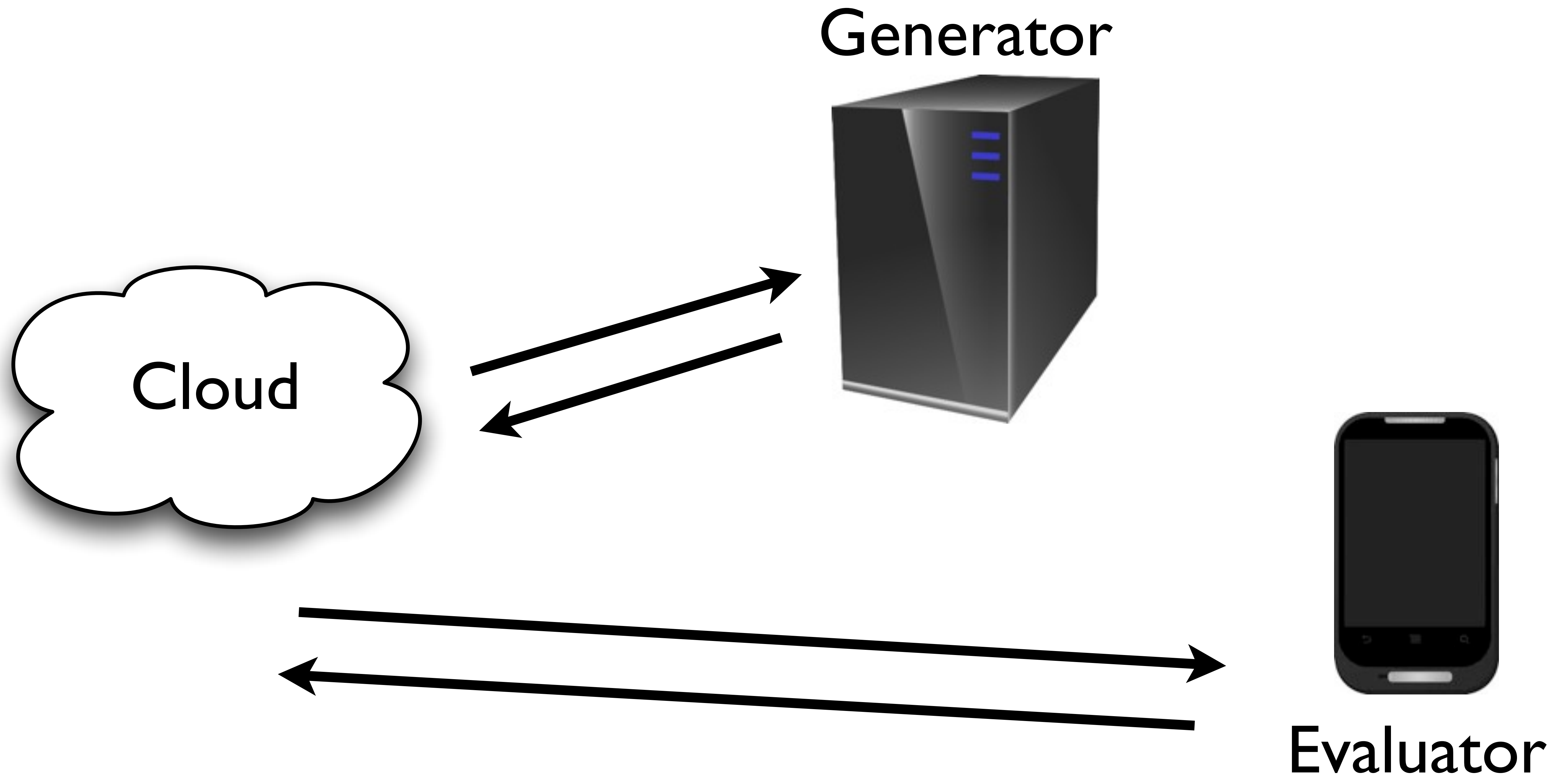
Circuit Evaluation

CMTB



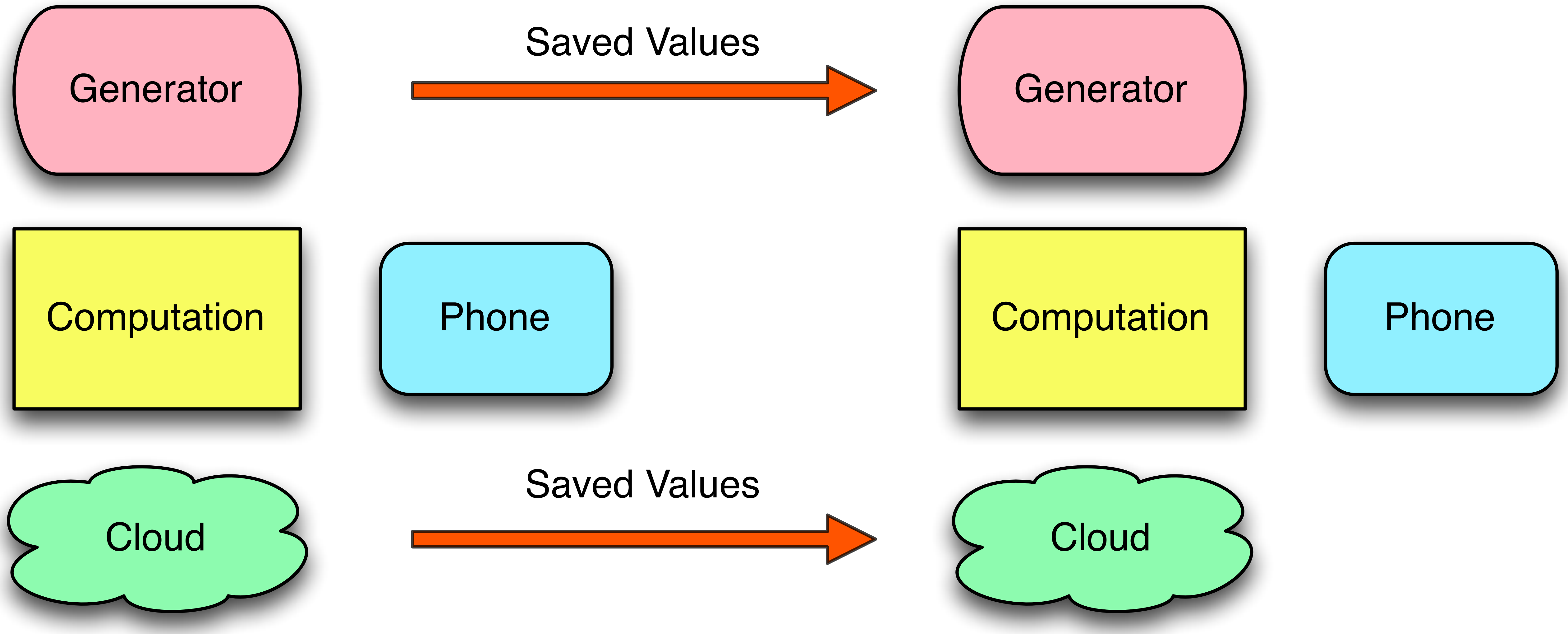
Circuit Commitment Check

CMTB



Output and Output check

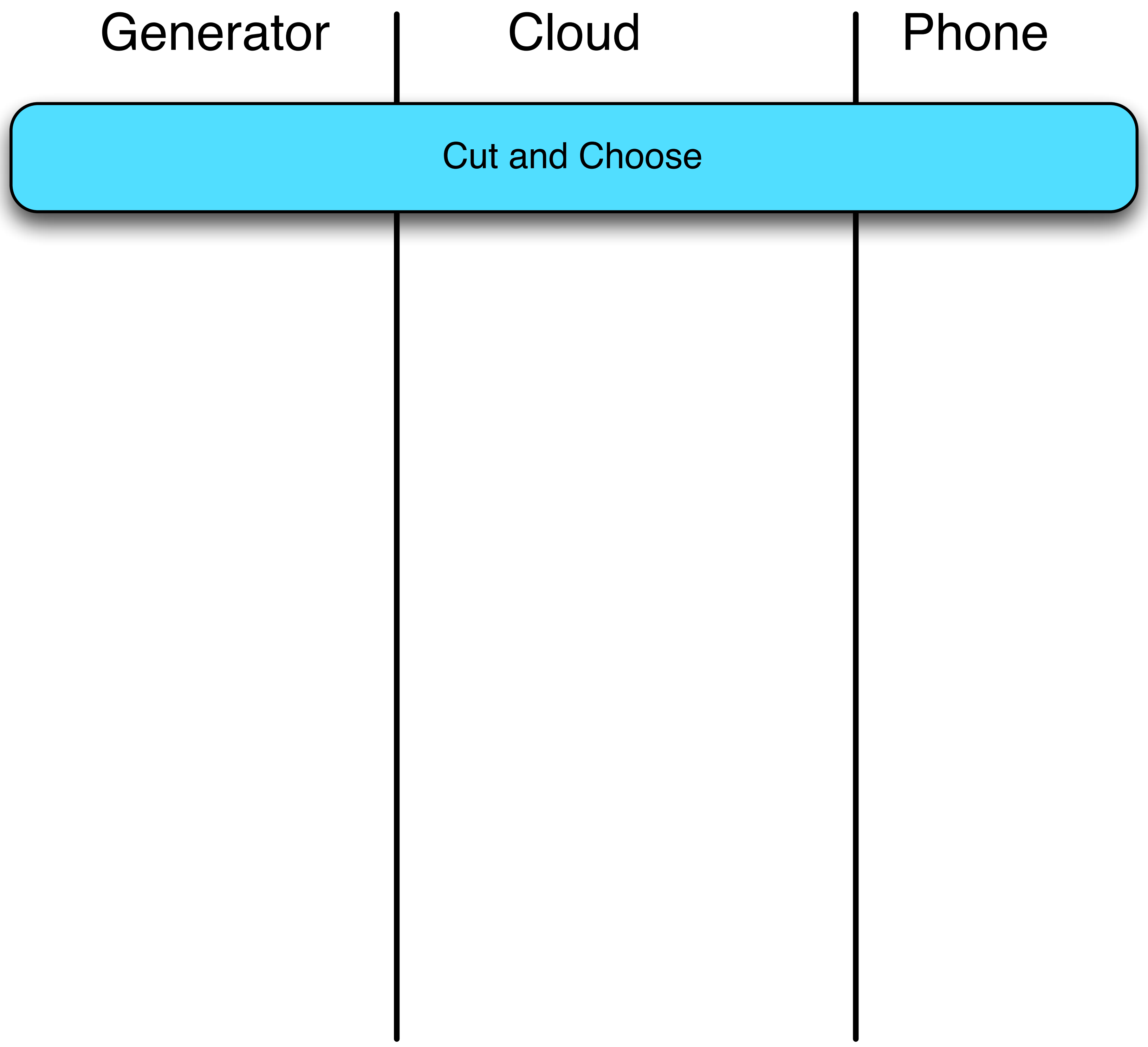
Outsourced PartialGC



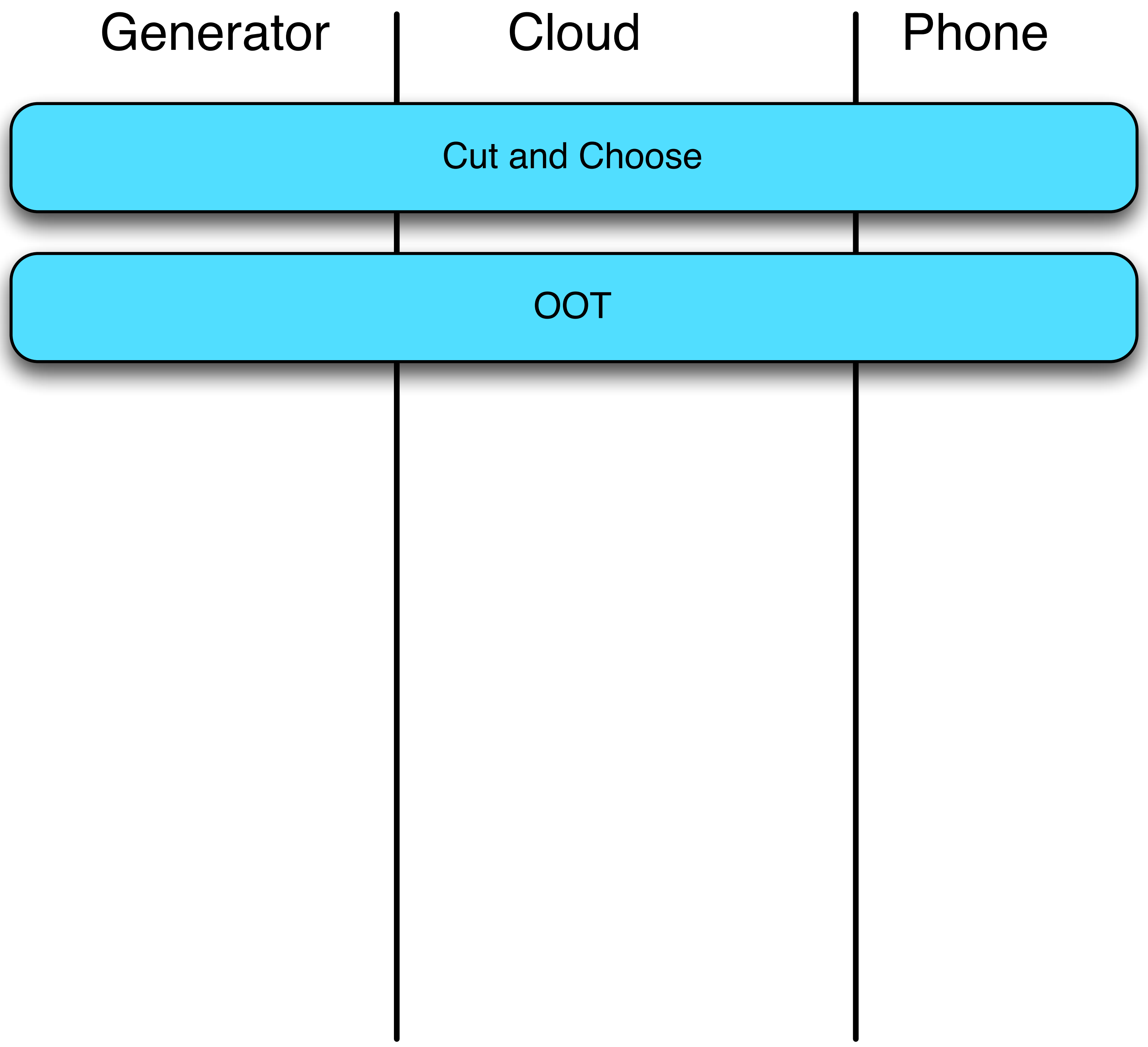
Protocol

Generator	Cloud	Phone

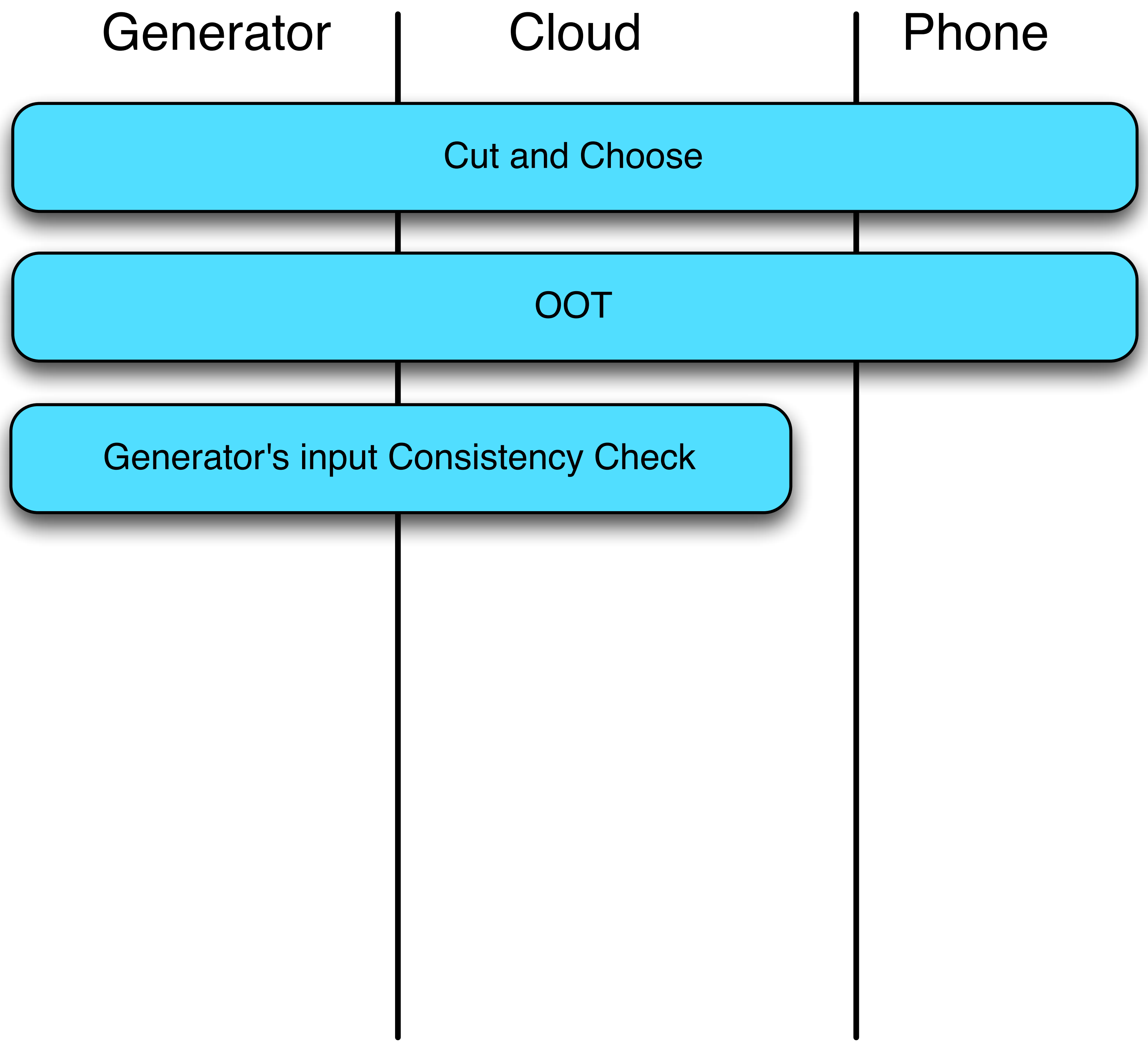
Protocol



Protocol

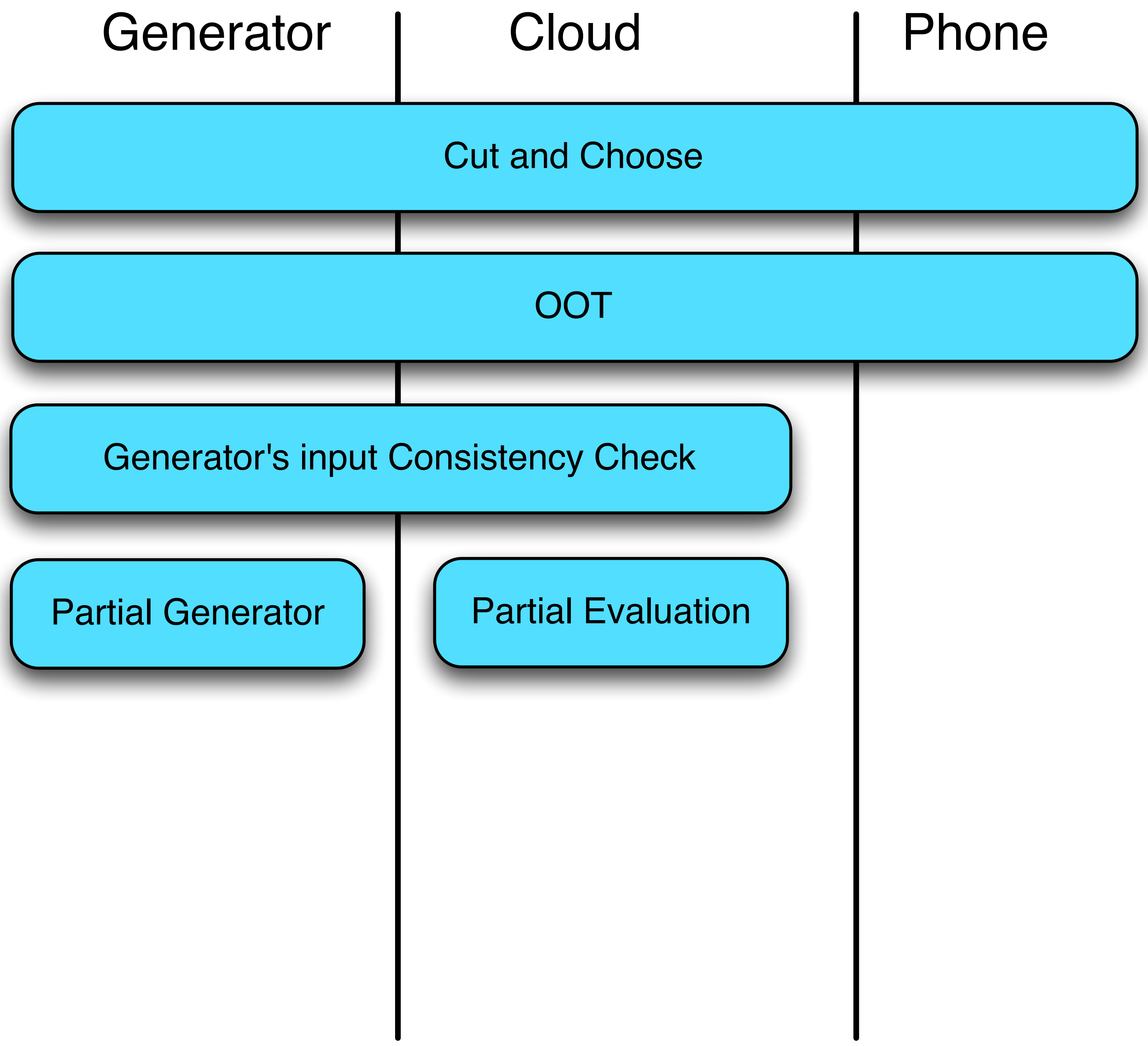


Protocol


Generator
Cloud
Phone
Cut and Choose
OOT
Generator's input Consistency Check

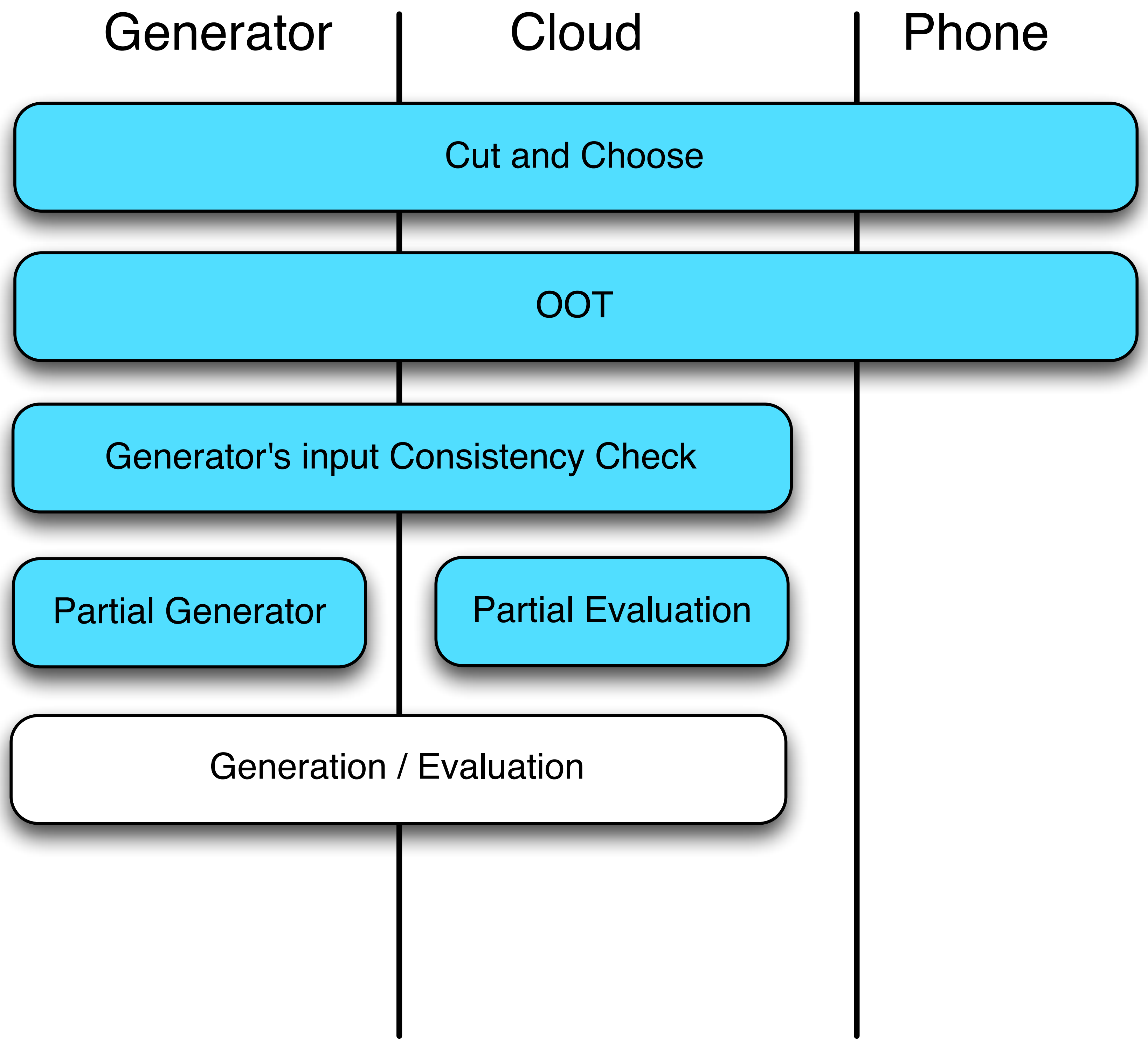
Protocol

UNIVERSITY OF OREGON

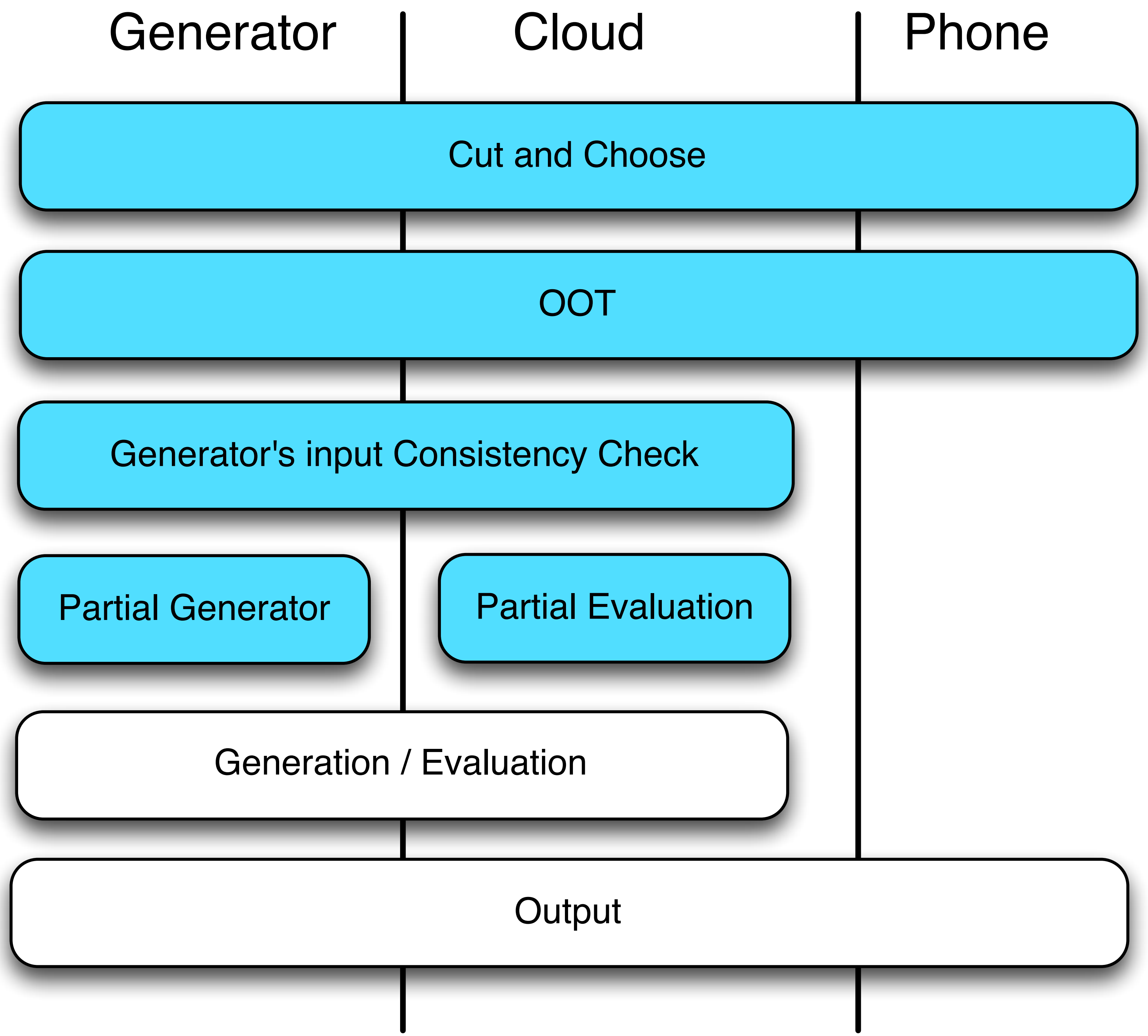


Protocol

UNIVERSITY OF OREGON

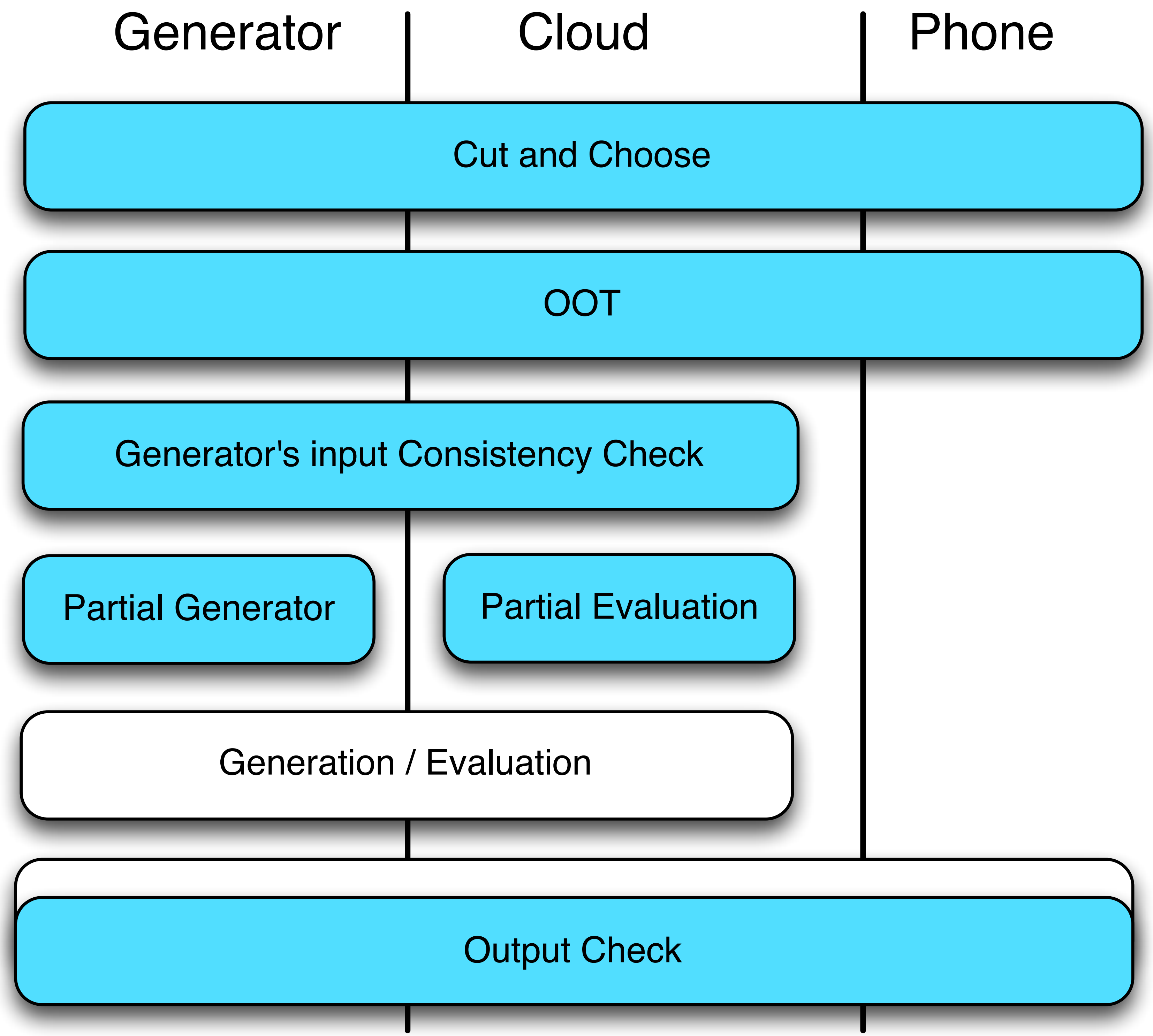


Protocol


Generator
Cloud
Phone
Cut and Choose
OOT
Generator's input Consistency Check
Partial Generator
Partial Evaluation
Generation / Evaluation
Output

Protocol

UNIVERSITY OF OREGON



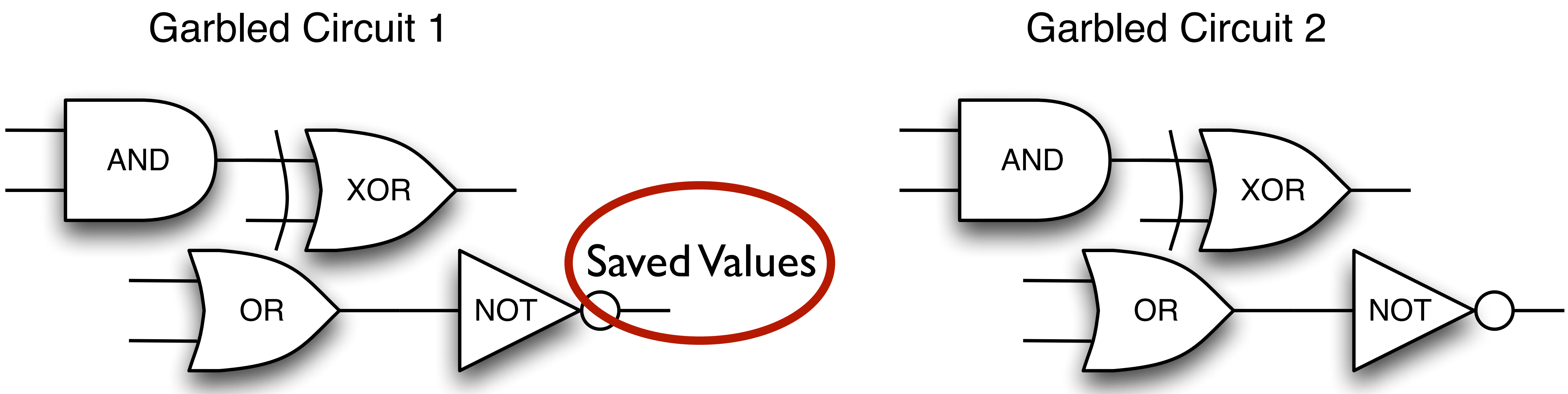
Output Check

- Output (x || MAC(x))
- Slower for circuit evaluation. Proof of concept implementation has ~14,000 non-XOR gates per 128 bits.
- Extremely fast for our outsourcing party [bits/128 MAC operations instead of the output proof with homomorphic XOR commitments].

Wrong Saved Values

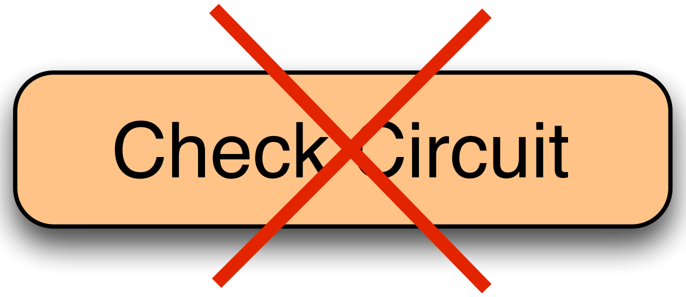
UNIVERSITY OF OREGON

- Generator gets caught through circuit check
- Cloud gets caught, assuming he continues in the computation, when the output check fails



Incorrect Check Circuits

- Aborting on incorrect check circuits gives away information about what circuits are check or evaluation.
- If check is found to be incorrect, then the remaining computation and any saved values must be abandoned.
- Cloud informs the Generator and Phone of the incorrect circuit and what it should have been.



Preliminary Test Results

UNIVERSITY OF OREGON

	64 Circuits			256 Circuits		
	CMTB	Partial		CMTB	Partial	
KeyedDB 64	$72 \pm 2\%$	$8.3 \pm 5\%$	8.7x	$290 \pm 2\%$	$26 \pm 2\%$	11x
KeyedDB 128	$140 \pm 2\%$	$9.5 \pm 4\%$	15x	$580 \pm 2\%$	$31 \pm 3\%$	19x
KeyedDB 256	$270 \pm 1\%$	$12 \pm 6\%$	23x	$1200 \pm 3\%$	$38 \pm 5\%$	32x

* both evaluated on same hardware, security parameters, and setup

Work in progress

UNIVERSITY OF OREGON

	64 Circuits			256 Circuits		
	CMTB	Partial		CMTB	Partial	
Largest Substring 128	$190 \pm 4\%$	$20 \pm 9\%$	9.5x	$800 \pm 7\%$	$84 \pm 9\%$	9.5x
Largest Substring 256	$370 \pm 4\%$	$40 \pm 10\%$	9.3x	$1700 \pm 8\%$	$130 \pm 7\%$	13x
Largest Substring 512	$730 \pm 4\%$	$70 \pm 10\%$	10x	-	$200 \pm 10\%$	-

- For comparison

-- In [CMTB] output and input values under a 1-time pad with MACs.

Conclusion

UNIVERSITY OF OREGON

- Saving wire labels
- Transform and check values
- Discussed our protocol and preliminary results
- Work in progess ...

Questions?

UNIVERSITY OF OREGON

Benjamin Mood, bmood@cs.uoregon.edu

OSIRIS Oregon Systems Infrastructure Research & Information Security Laboratory

http://osiris.cs.uoregon.edu/

Questions?

UNIVERSITY OF OREGON

Benjamin Mood, bmood@cs.uoregon.edu

OSIRIS Oregon Systems Infrastructure Research & Information Security Laboratory

http://osiris.cs.uoregon.edu/