


IC Piracy Prevention via Design Withholding and Entanglement

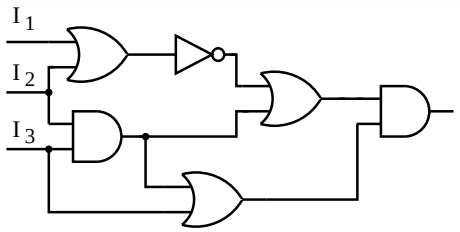
Soroush Khaleghi
Kai Da Zhao
Wenjing Rao

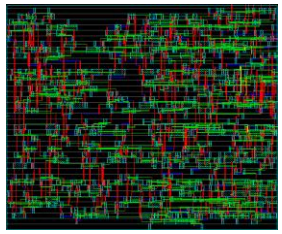
UIC UNIVERSITY OF ILLINOIS AT CHICAGO
Department of Electrical and Computer Engineering
COLLEGE OF ENGINEERING

IC Piracy & Reverse Engineering

Horizontal

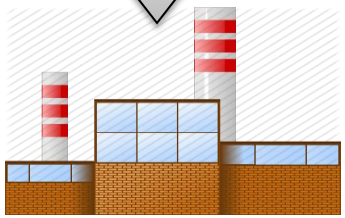
```
module DUT (A, B, S, out);
input A, B, S;
output reg out;
always @(*)
begin
  if (S==1)
     out = A;
  else
     out =B;
end
endmodule
```



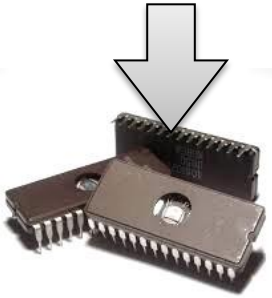


HDL → Synthesis → Placement & Routing

Untrusted Manufacturer

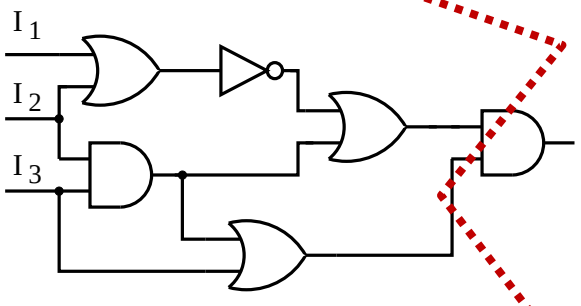


Fabrication



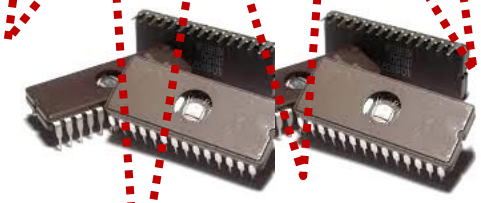
Reverse Eng.:

Steal design information

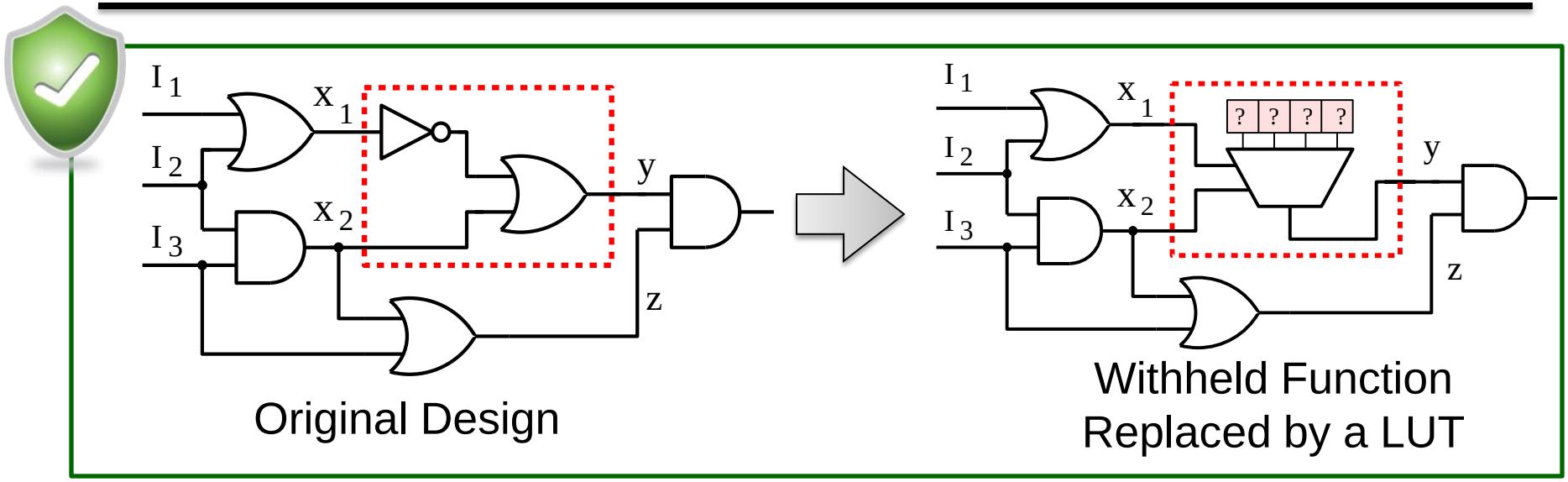


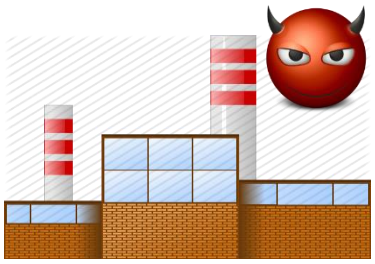
IC Piracy:

Fabricate more chips



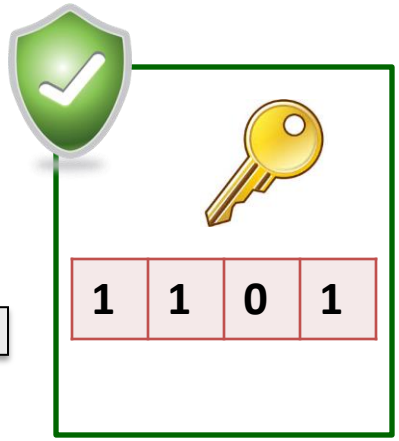
Design Withholding Flow



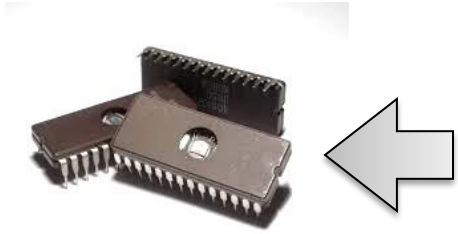


Manufacturer

Locked Chips

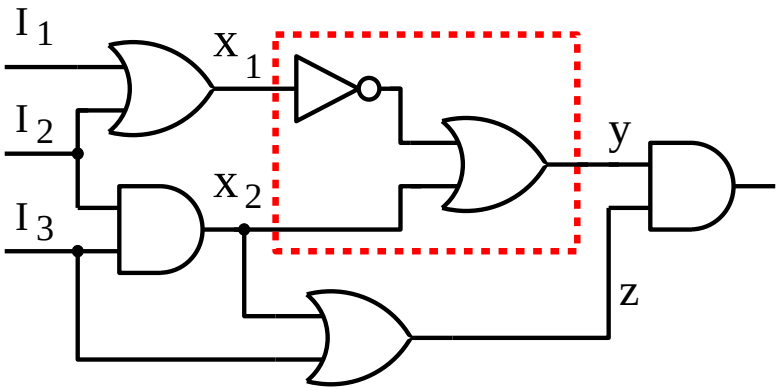


Trusted Facility



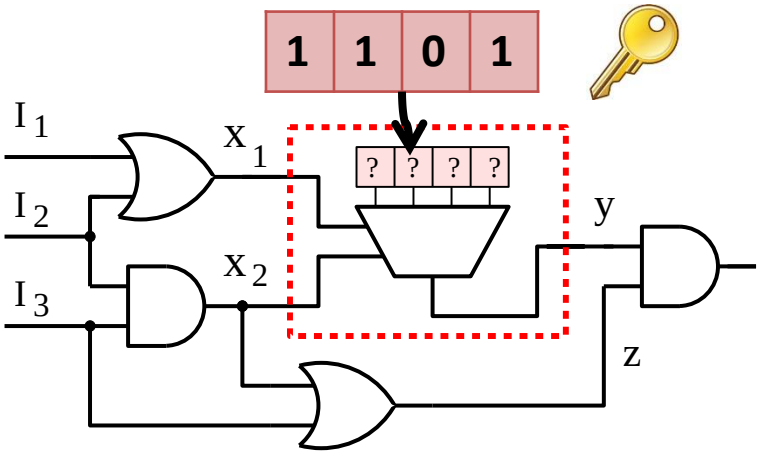
Open Market
(Activated Chips)

Design Withholding Mechanism

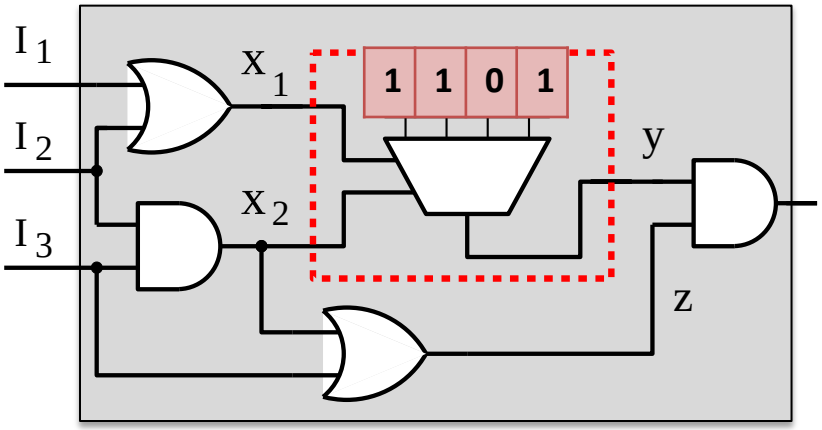


$$y = x_2 + x_1'$$

x_2	x_1	y
0	0	**1**
0	1	**0**
1	0	**1**
1	1	**1**



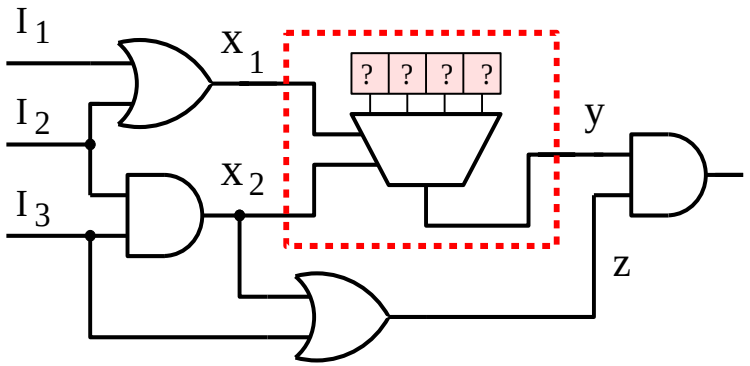
Working Chip

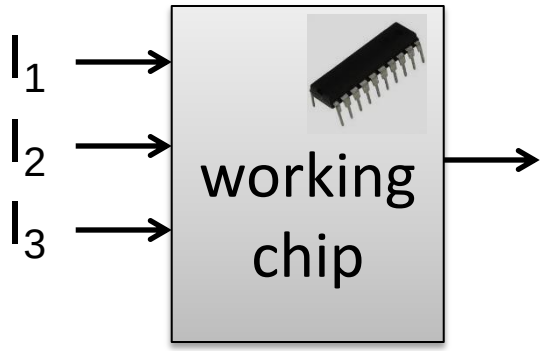


Attacker's Tools and Goal

Tools / Resources:

- ❑ Partial design file
- ❑ A working chip
- ❑ Tools & capabilities
  - Simulation tools
  - Ability to modify the design
  - Fabrication





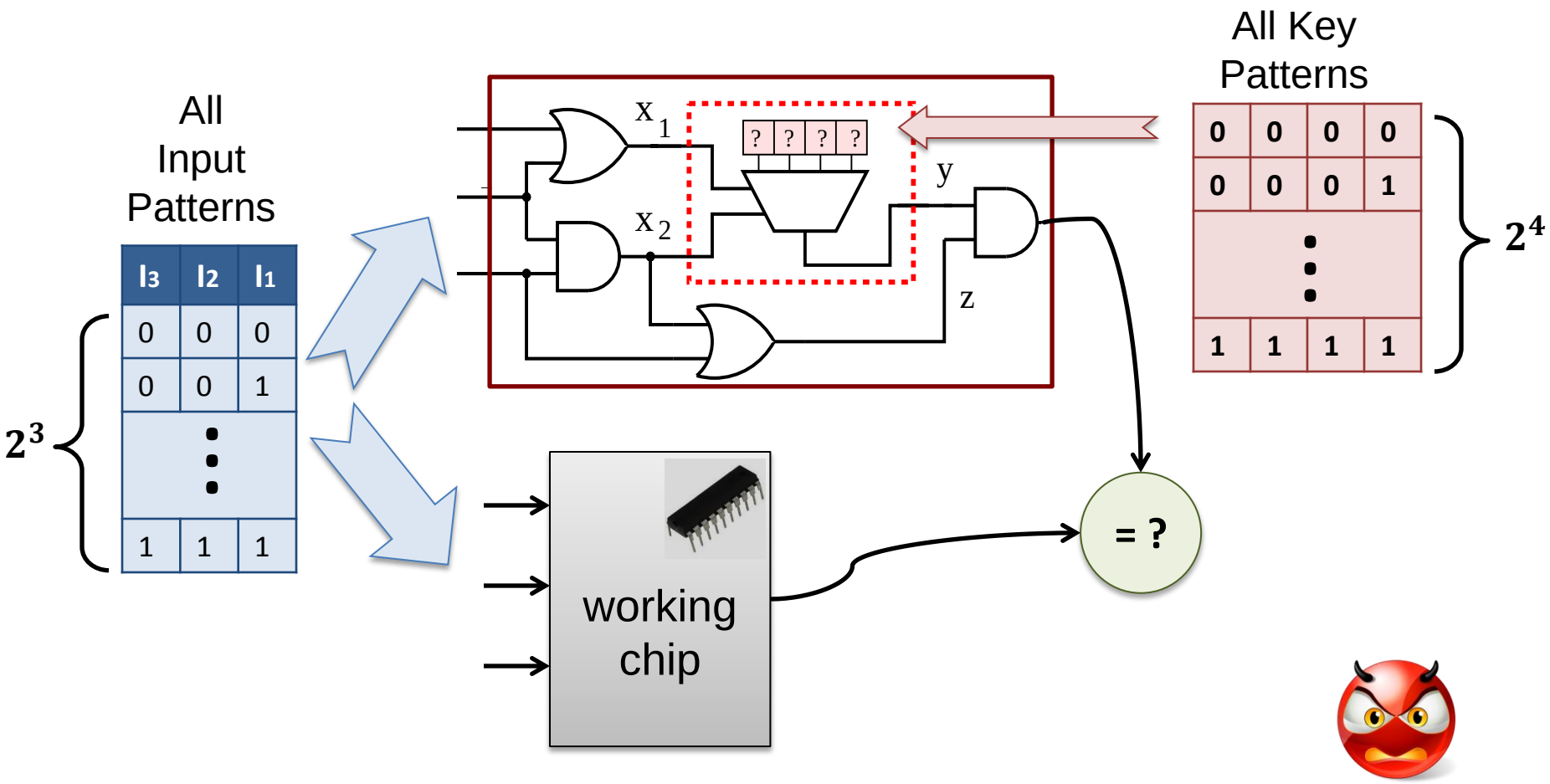
Goal:

❖ Recover the **KEY:** content of LUT (the withheld function)

➢ IC Piracy and Reverse Engineering

1	1	0	1

Brute Force Attack

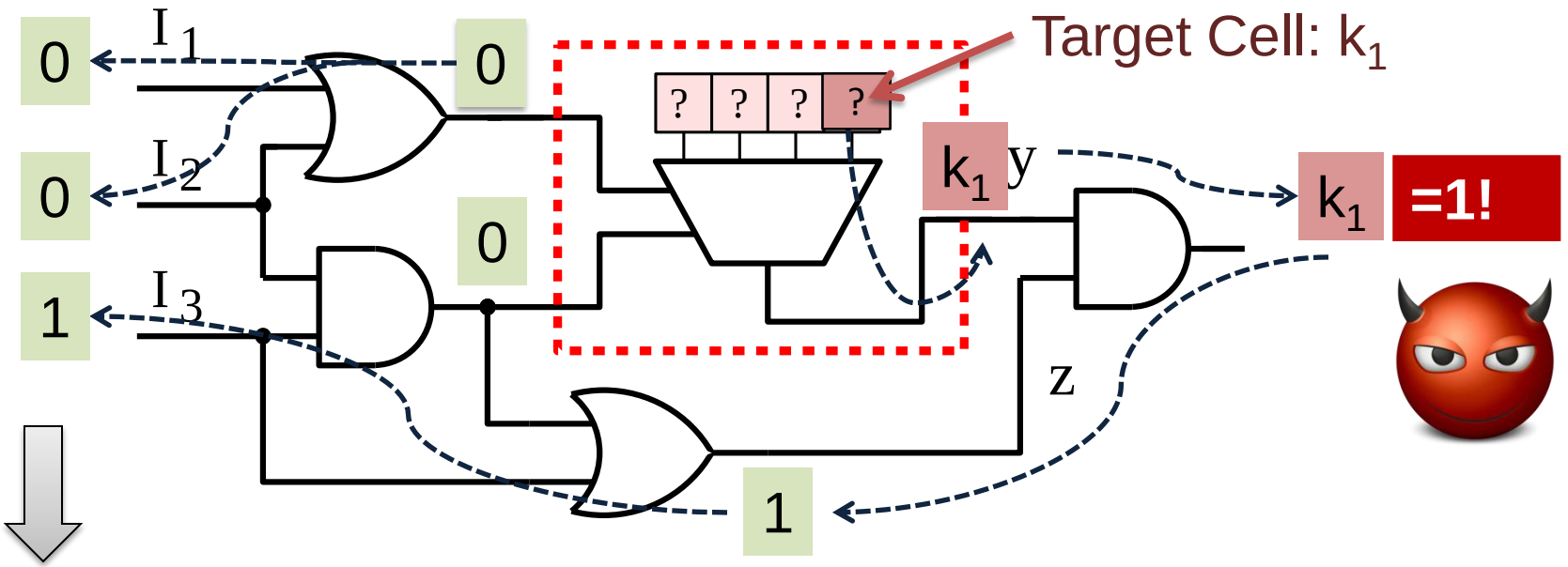


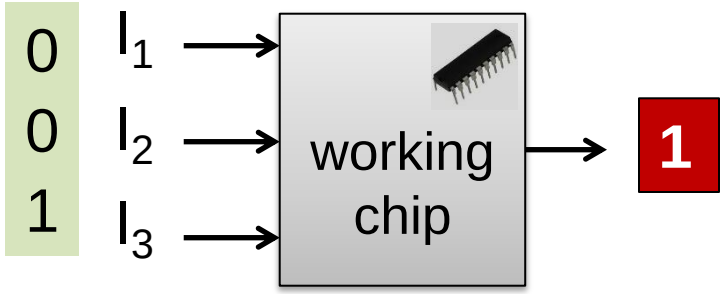
- Number of inputs: I
- Key size: K

Attack Complexity: $O(2^{K+I})$

A More Powerful Attack:

- Solving the key, **one cell at a time**!
- Finding input to isolate and propagate a key cell's value to output

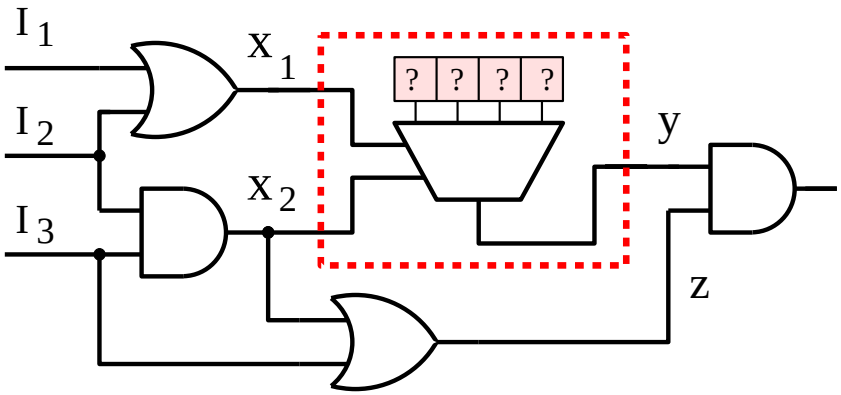




- = Automatic Test Pattern Generation in VLSI testing
- Powerful tools/algorithms available

Runtime Comparison:

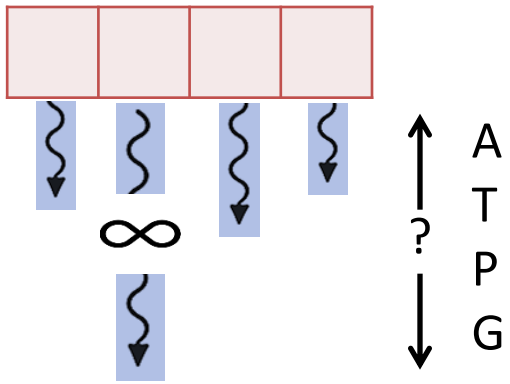
- Number of inputs: p
- Key size: k



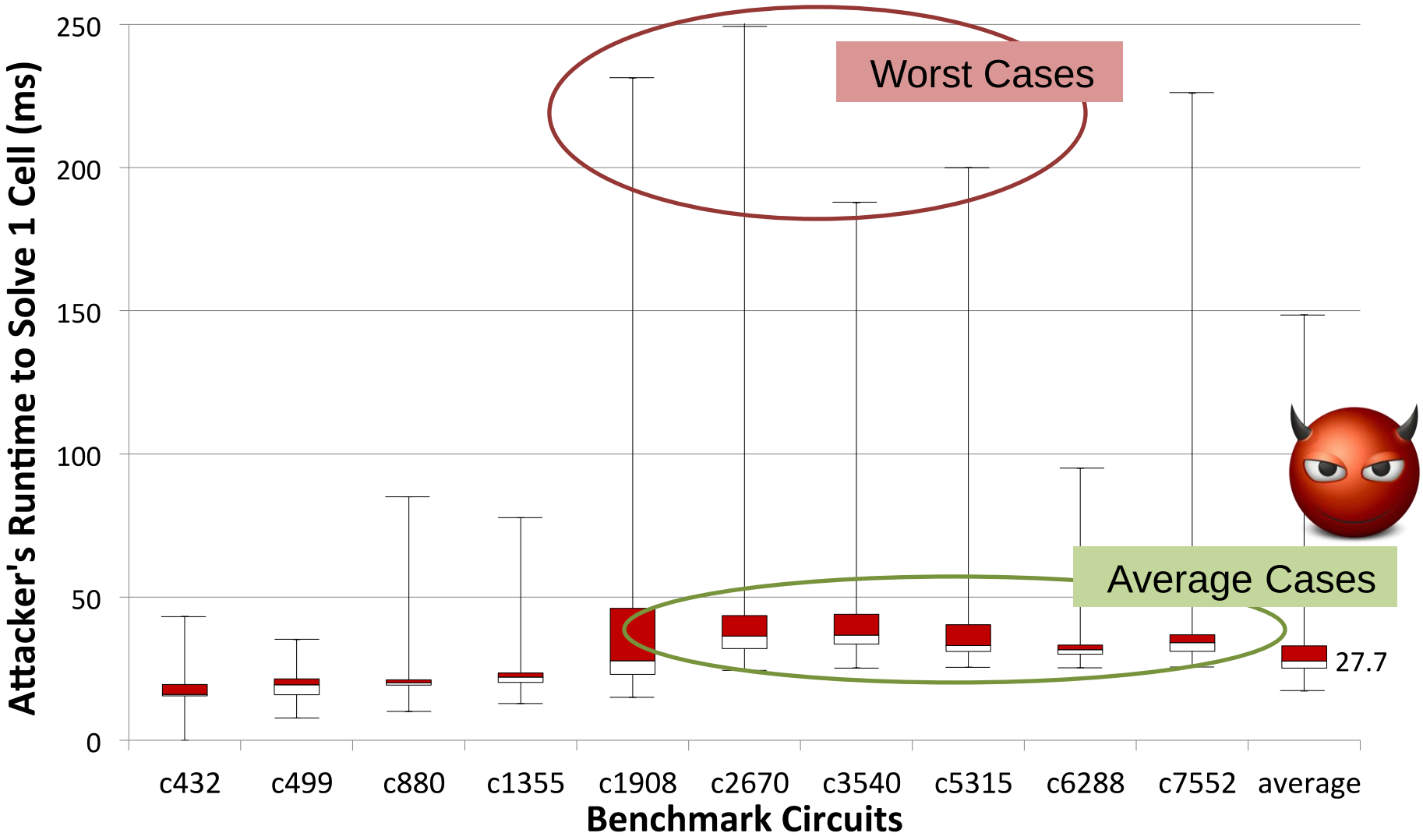
Worst-case Scenario:

- Brute-Force Attack: $O(2^{p+k})$
- ATPG-based Attack: $O(t * k)$

t:
Time to solve an ATPG problem:
NP-Complete

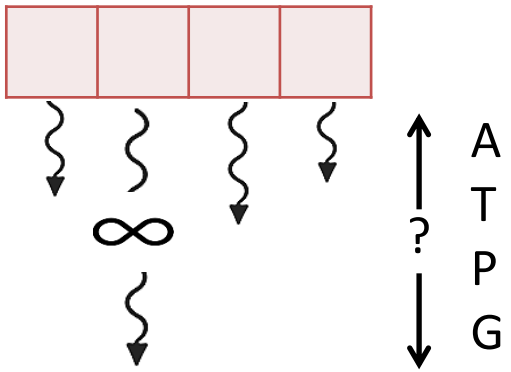


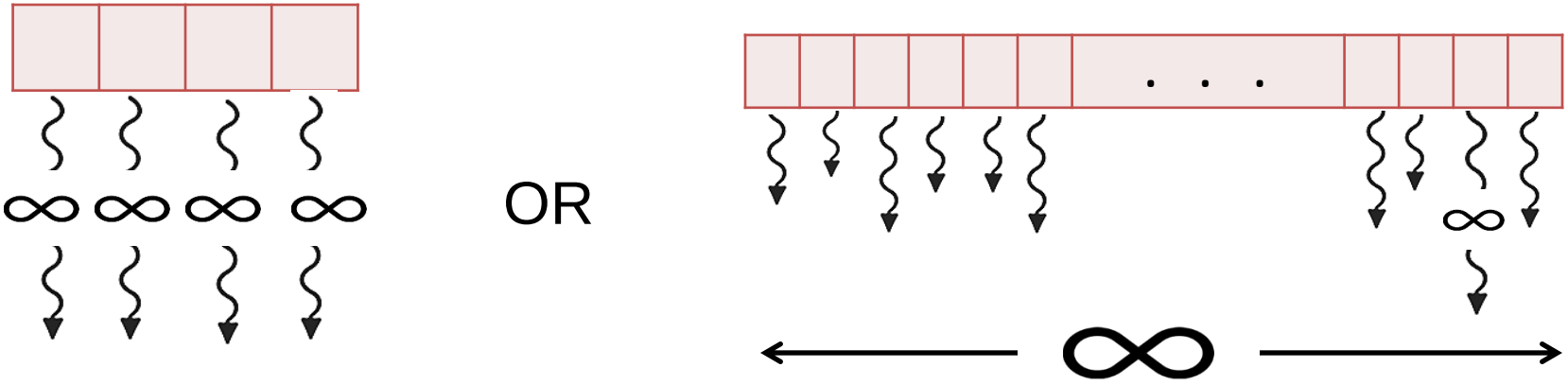
Solving an ATPG Problem - runtime:



A Strong Defense Framework

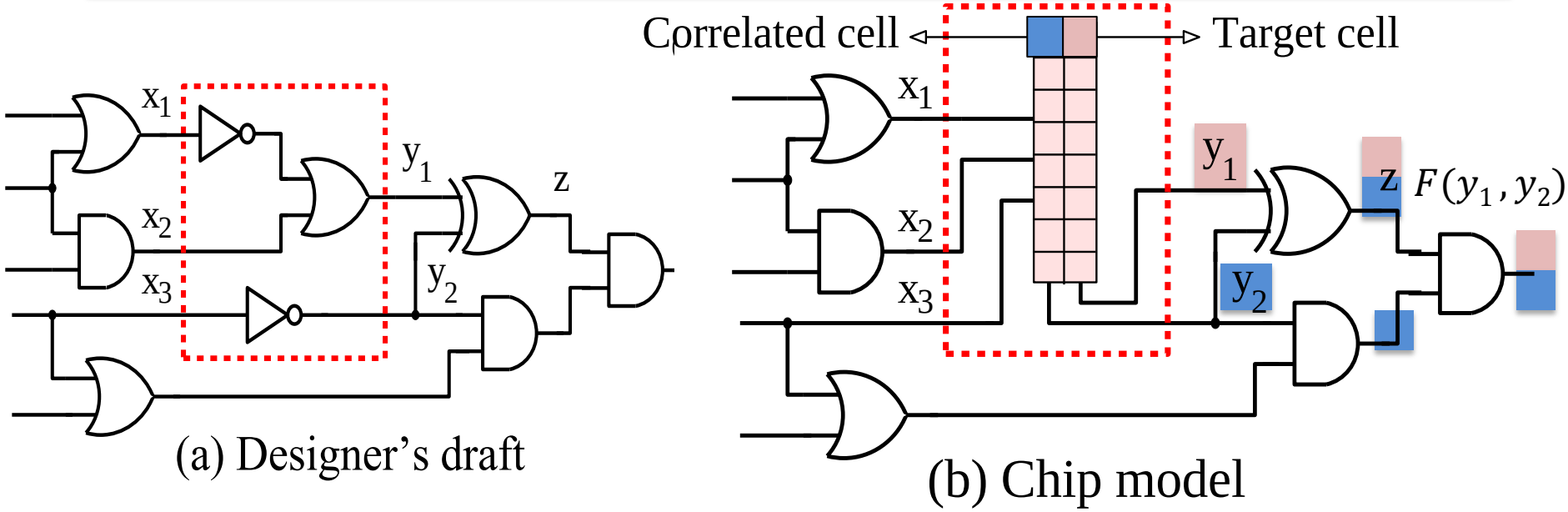
- Design Withholding's defense analysis



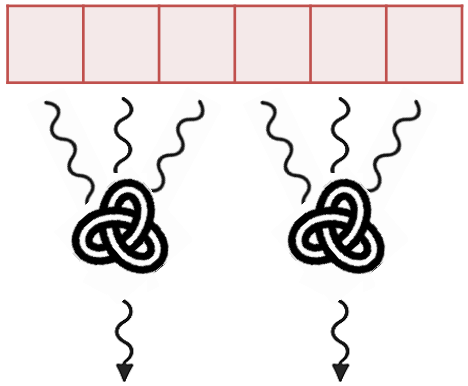


1) **Exponentially harder** problems
2) **Exponentially enlarged** key size

Motivation: Correlated Cells

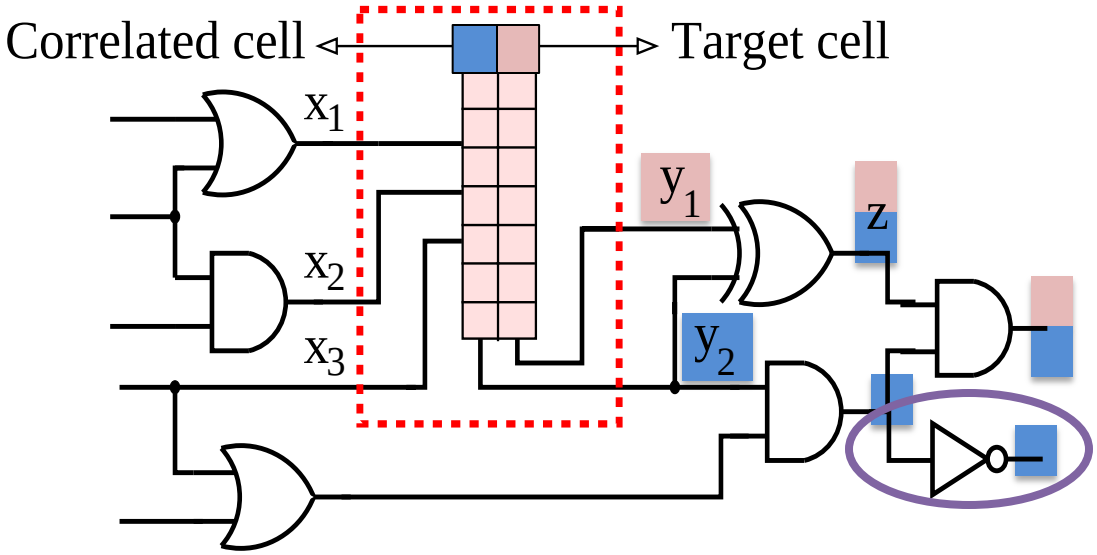


- Impossible to avoid the interference of a correlated cell's value
- Analogous to ATPG with *unknowns:* **NP-Hard**
- Gets even harder as the # of correlated cell ↗
  - Symbolic manipulation of large # of variables
  - Brute-Force



A Need to "Engineer" Correlated Cells

1) "Leaking" cases that might break the correlation



2) Hard to find a "perfect" function with guaranteed # of correlations

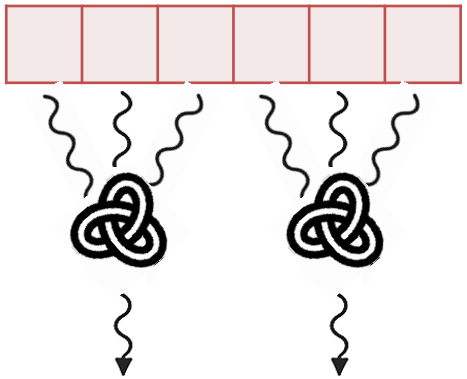
Overview of the Proposed Scheme

- **Entanglement**:
  - Systematically build correlations
  - Guaranteed # of key cells to be correlated
- **Outcomes:**

Attack Choice 1:

Symbolic Manipulation

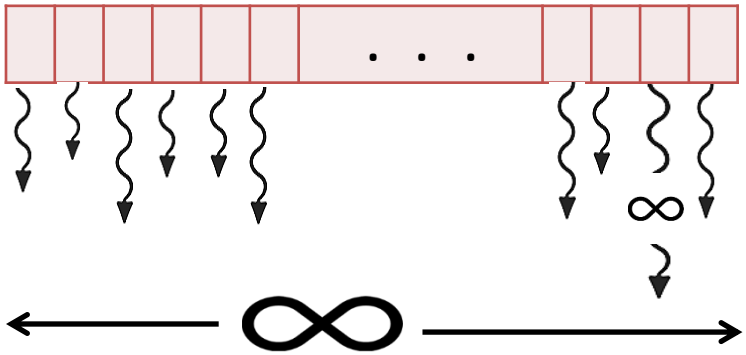
Exponentially harder problems



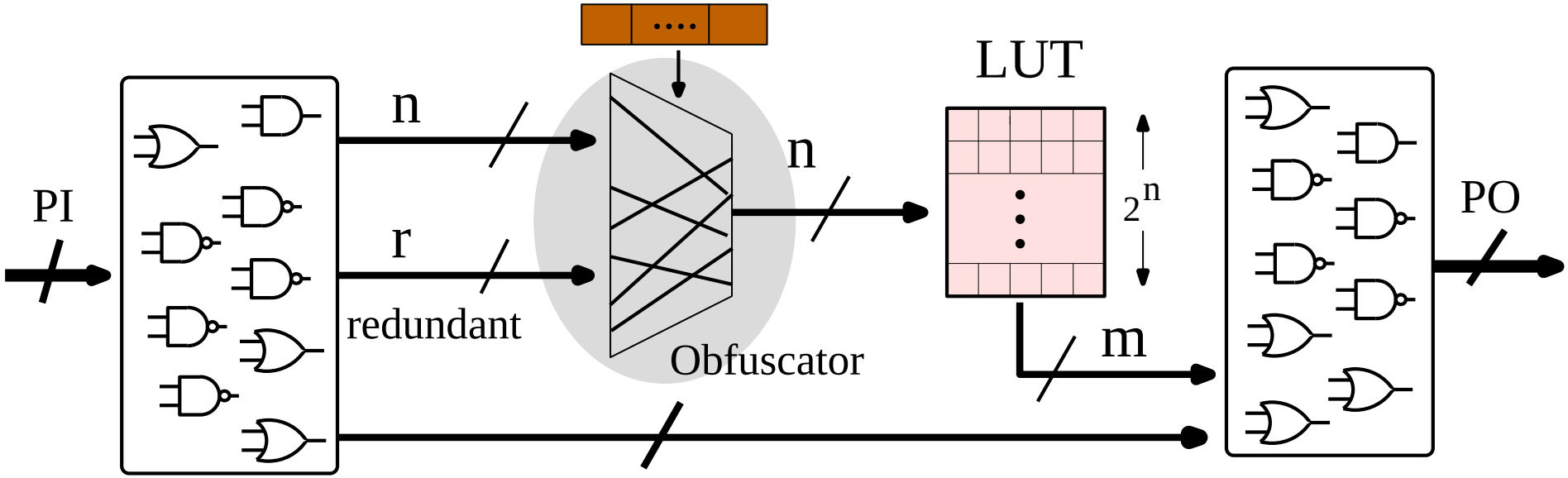
Attack Choice 2:

Modeling an enlarged LUT

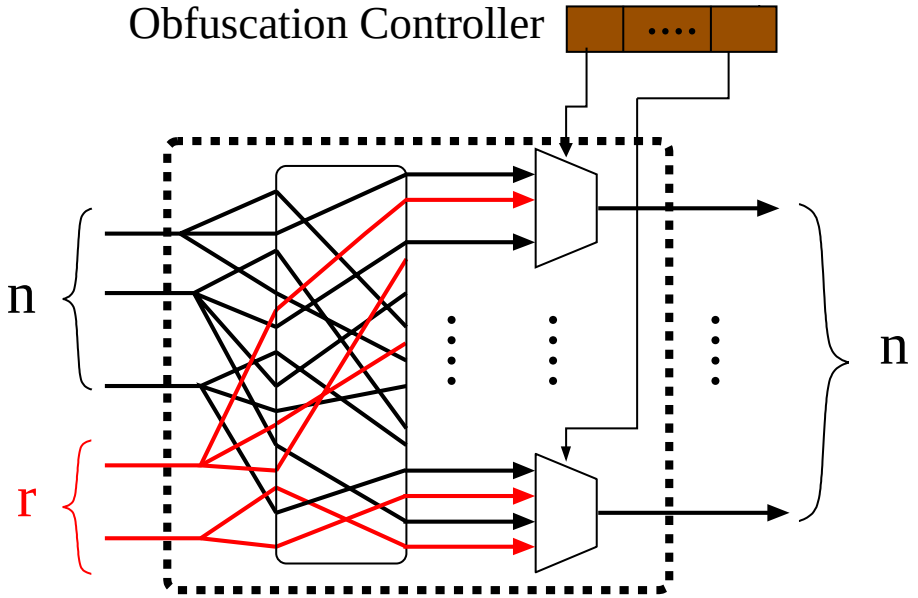
Exponentially enlarged key size



Entanglement via an Obfuscator

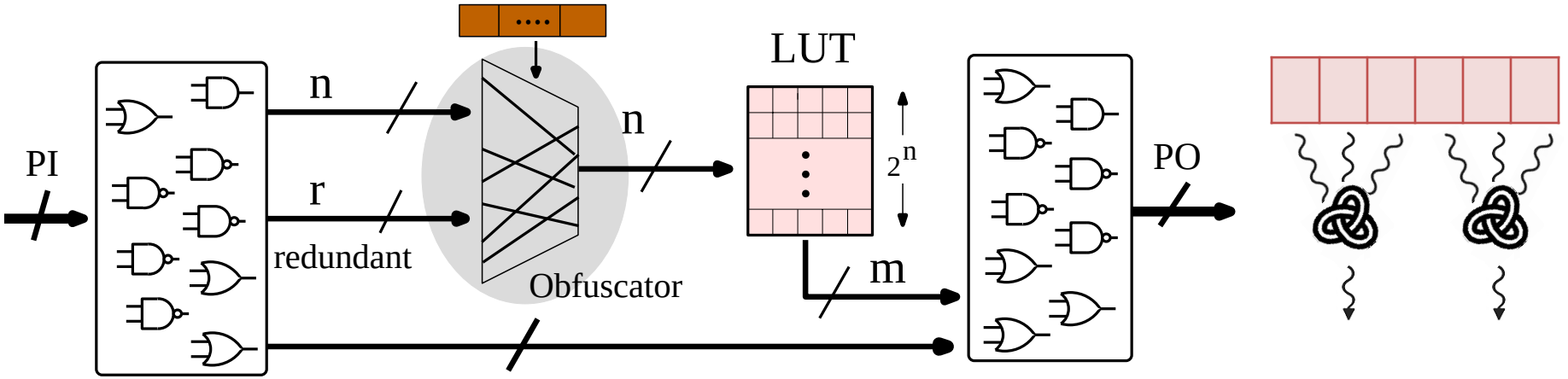


- **Obfuscator**
  - noise signals (redundancy)
  - original vs. redundant signals
  - blocks the redundant signals
  - Needs some key

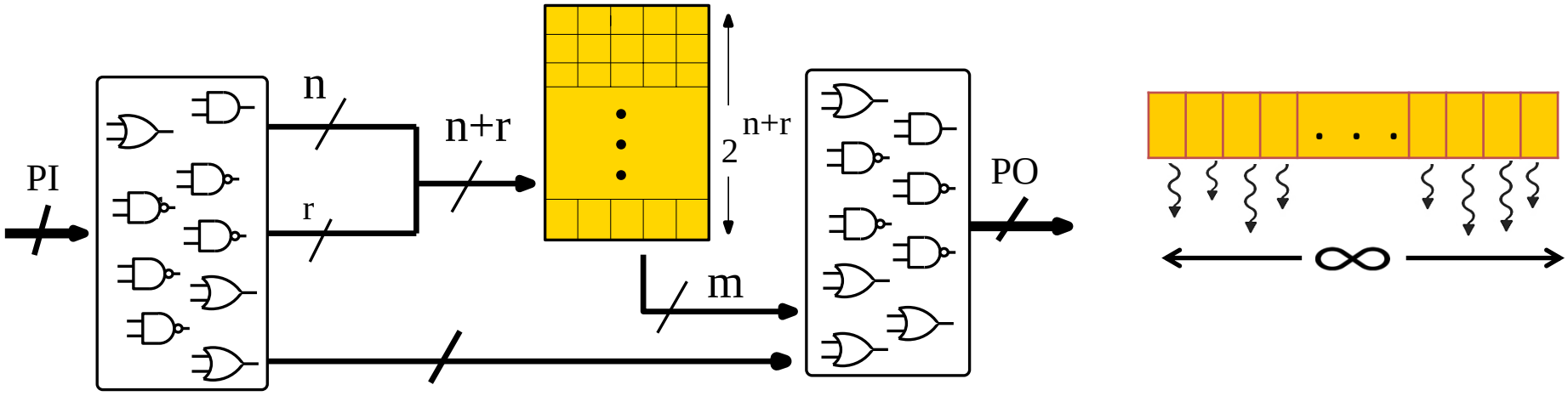


Attacker's View and Choices

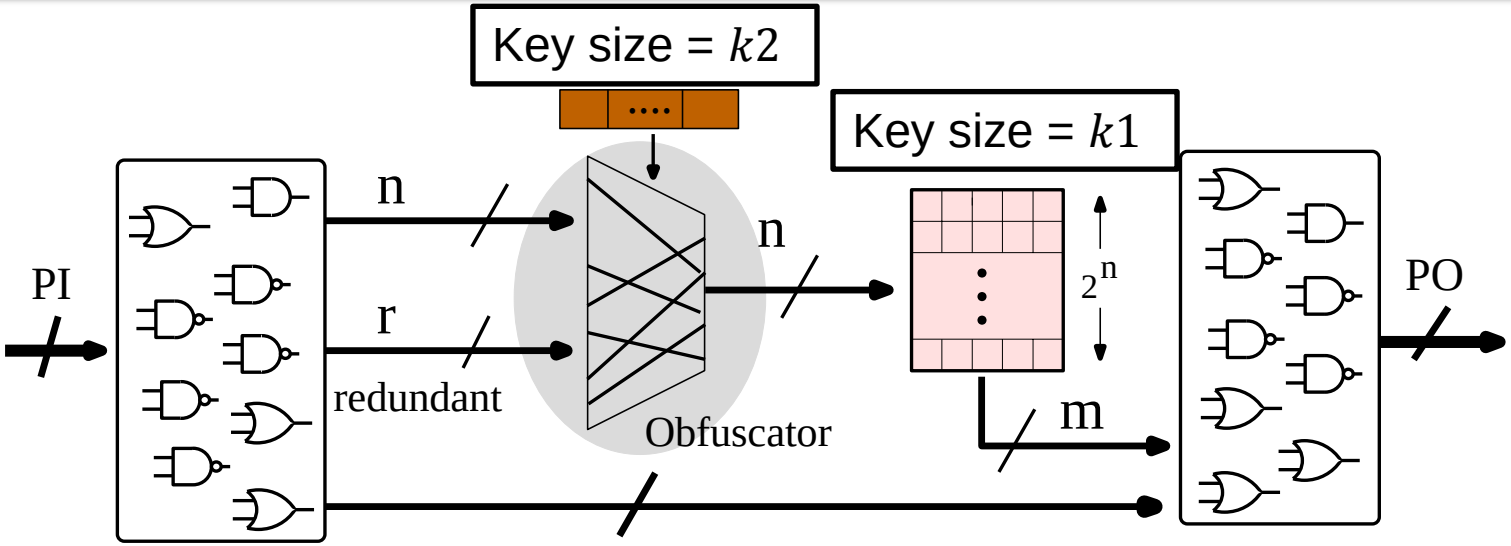
1) Choice 1: Symbolic Manipulation



2) Choice 2: Enlarged LUT



Cost Analysis



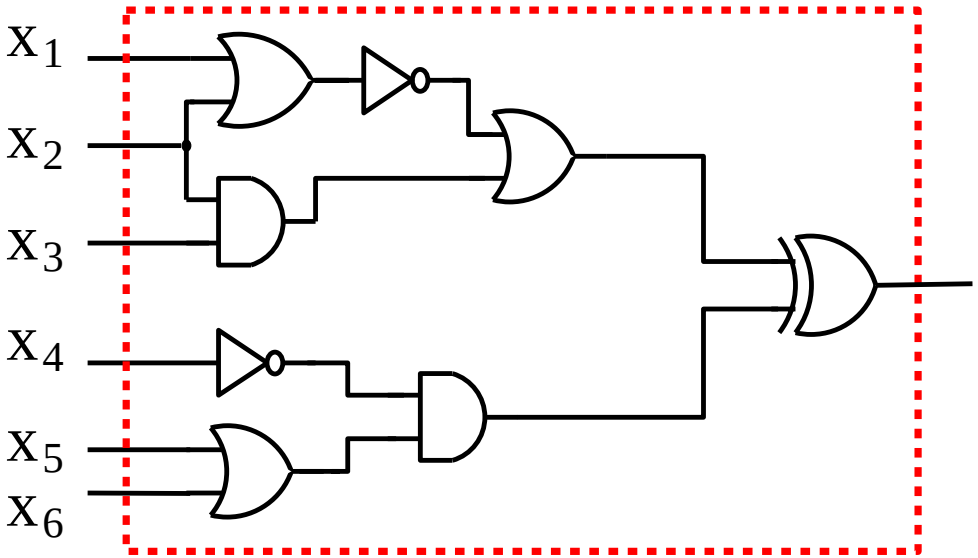
Hardware Cost Physical LUT **+** Obfuscator	$O(k1 + k2)$
Attack 1 Symbolic Manipulation	$O(2^{k1+k2})$
Attack 2 Enlarged LUT	$O(k1 * 2^{2^{k2}})$

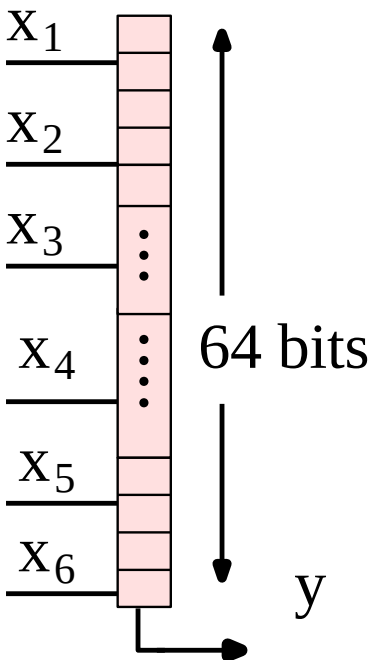
Hardware Cost →

Attacker's Cost ↗

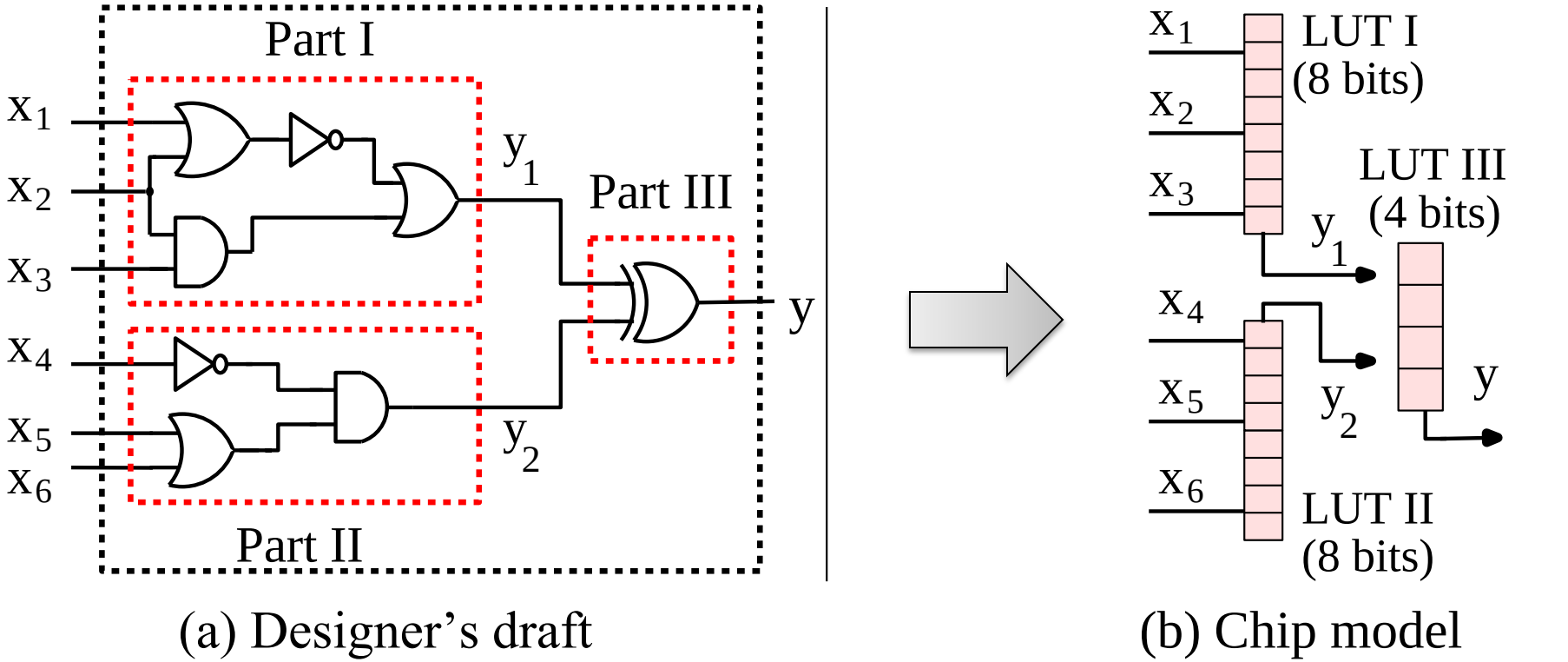
“Split & Entangle them”

- **External Entanglement: add keys and entangle them**
  - Small withheld function, + overhead
  - Attack complexity ↗ ; designer’s cost →
- The other way around? **Internal Entanglement**…
  - Split a big function to be withheld, **-** cost
  - Designer’s cost ↘ ; attacker’s cost →





Internal Entanglement



(a) Designer's draft

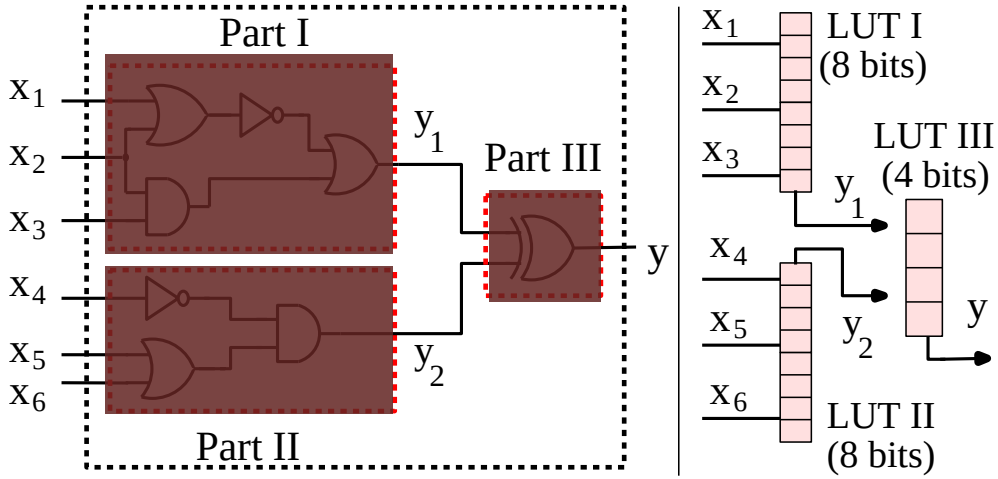
(b) Chip model

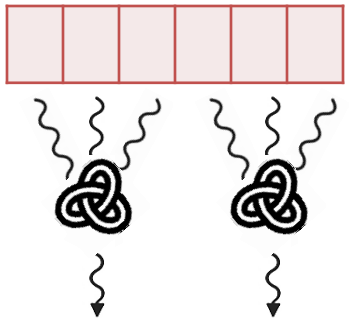
- The design is divided into **two layers**:

  output of LUTs in the first layer are the address lines of the LUT in the second layer

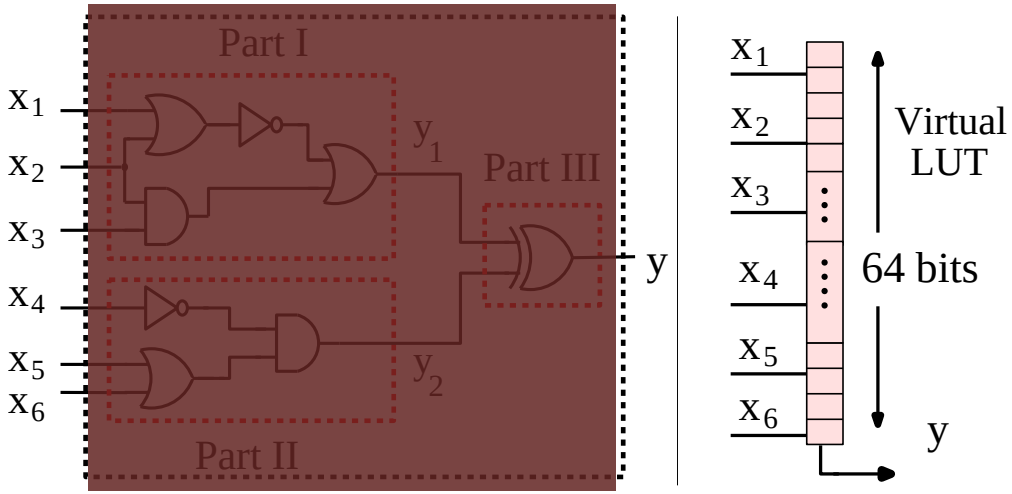
Attacker's View and Choices

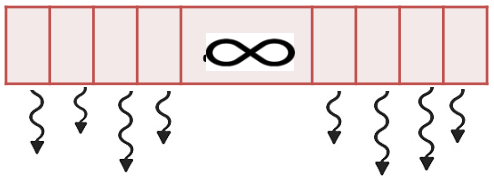
1) Choice 1: Symbolic Manipulation



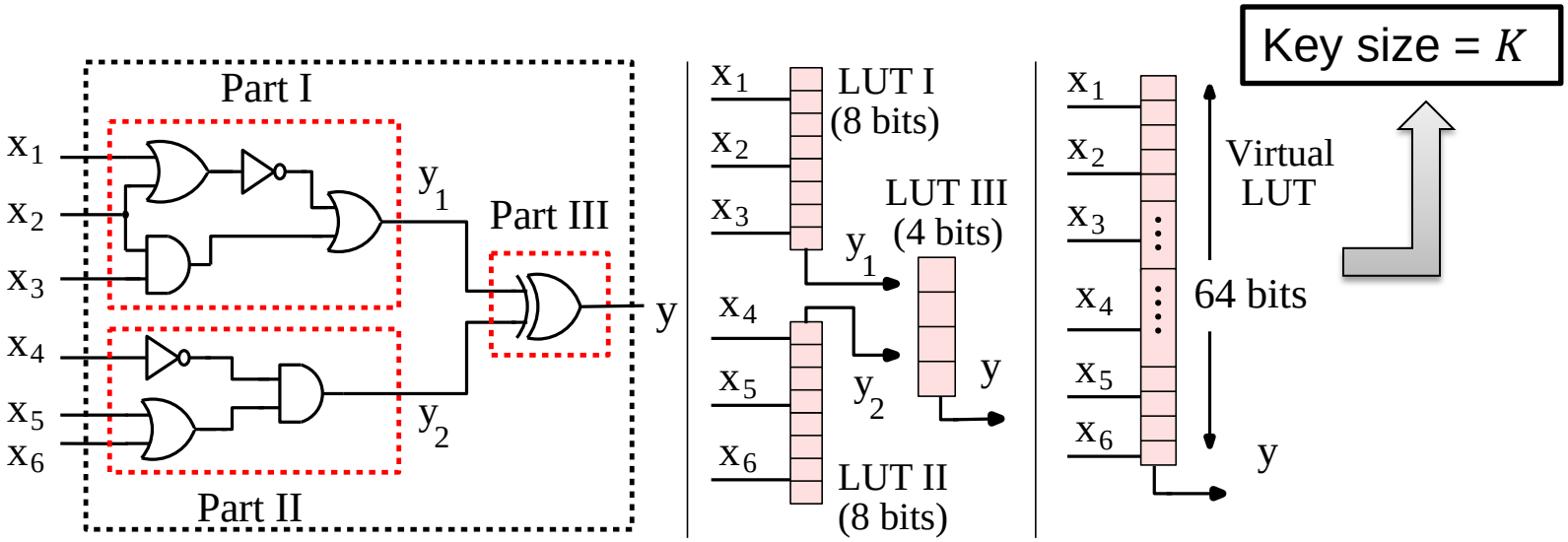


2) Choice 2: Enlarged LUT





Cost Analysis

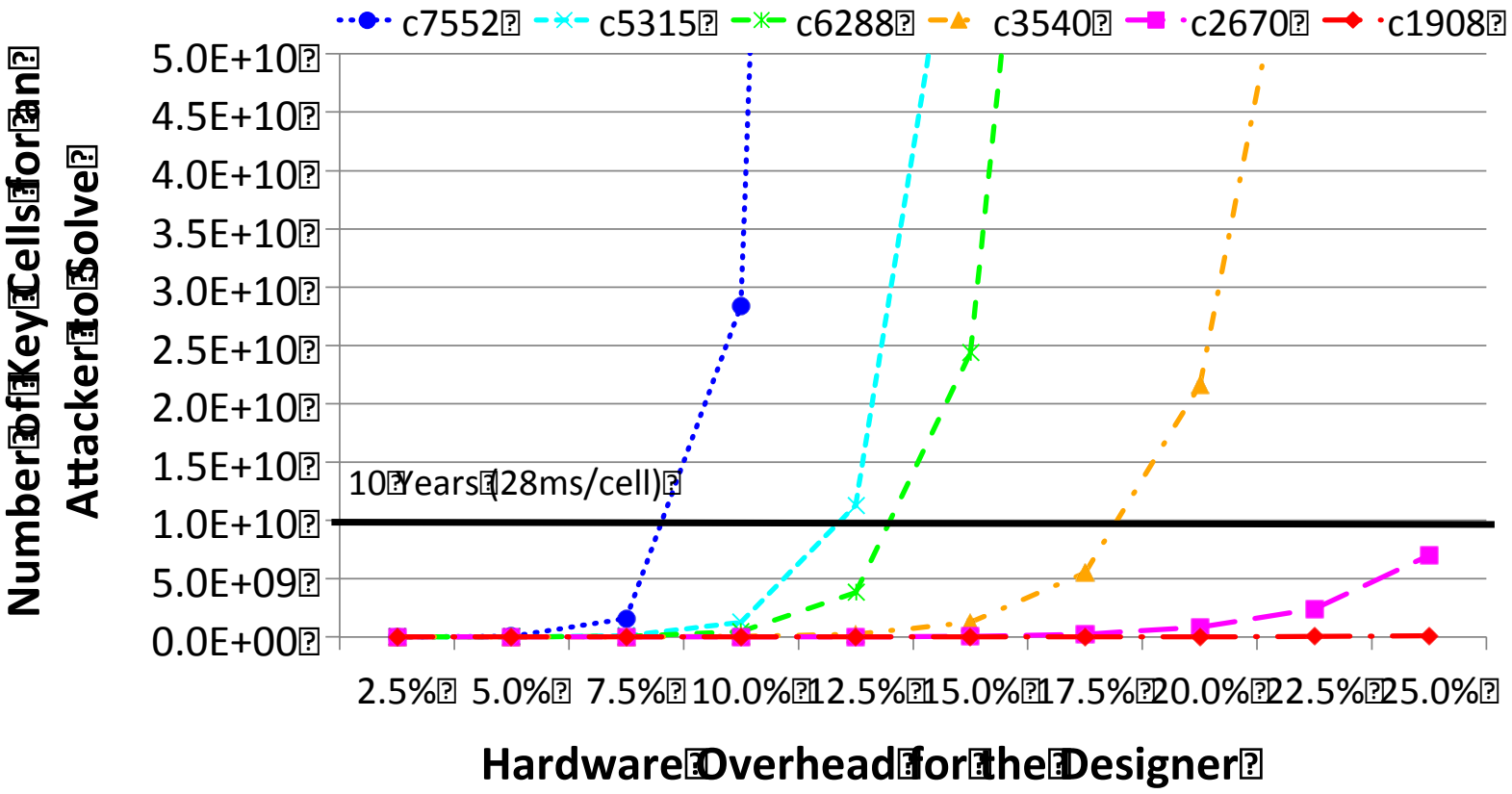


Hardware Cost Physical LUT	$O(\log K)$
Attack 1 Symbolic Manipulation	$O(K)$
Attack 2 Enlarged LUT	$O(K)$

Key size = 20

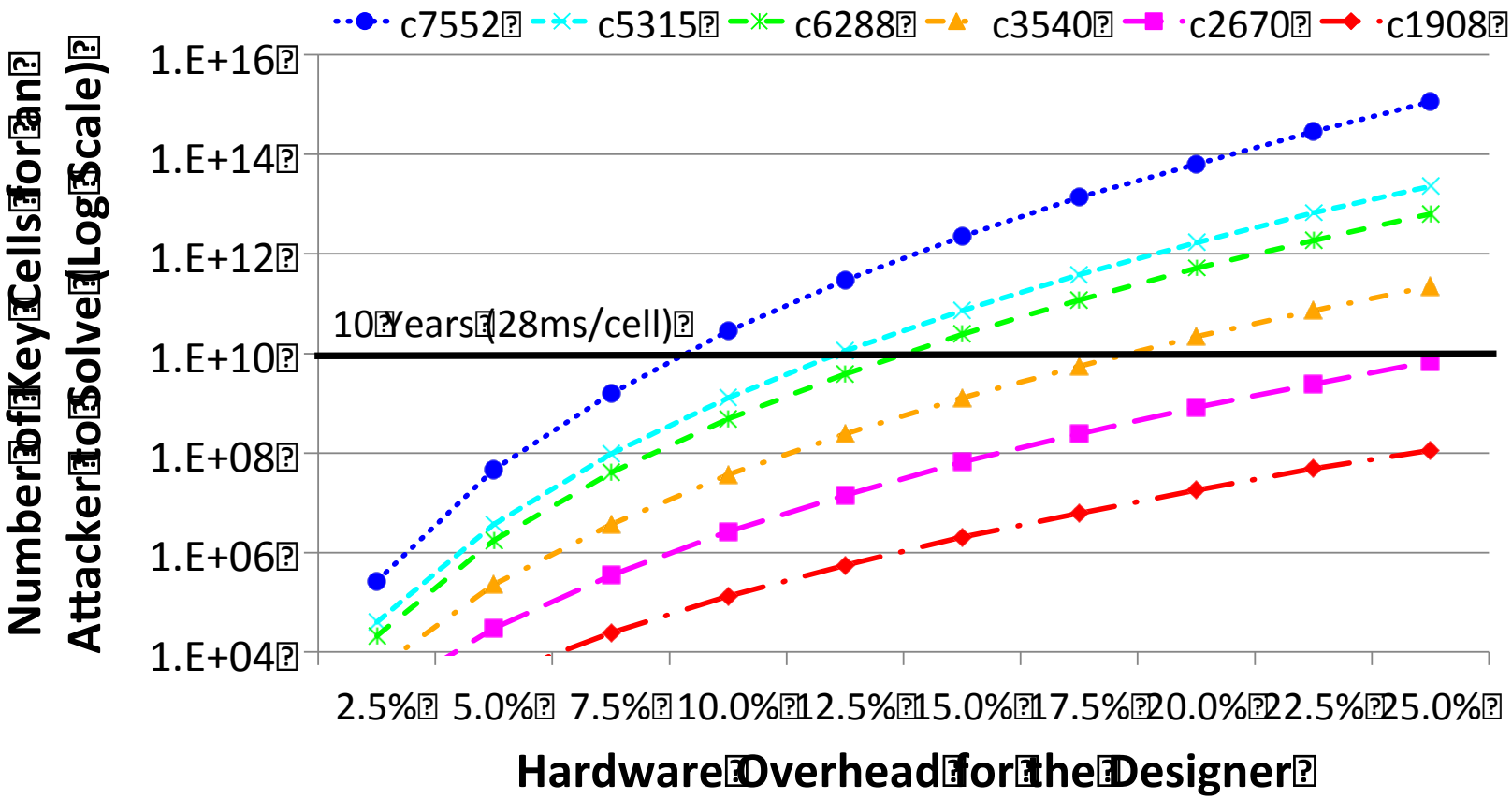
Key size = 64

Results



- Internal Entanglement Scheme
- Attack Complexity vs. Hardware Overhead (% transistors)
- 10 years line, based on average time per cell

Results



- Attack Complexity vs. Hardware Overhead
- Logarithmic Scale
- Scalability of the scheme

Conclusions

✧ IC Piracy and Reverse Engineering

✧ ATPG-based attack : powerful and fast

- NP-Complete cannot be relied for defense

✧ Entanglement

- Systematically correlate key cells

✧ Possible Attacks

1) Small number of exceedingly hard problems → Brute-Force
2) Hugely boosted number of ATPG problems

✧ Low hardware overhead at designer's side

✧ Secure and Scalable