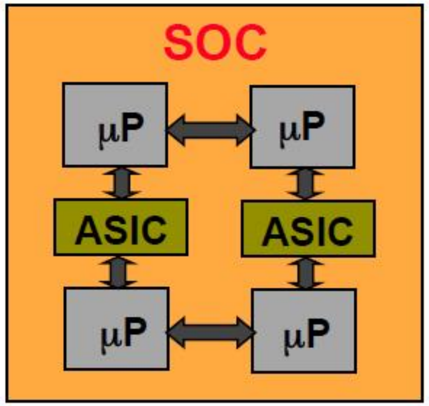
Hardware IP Protection

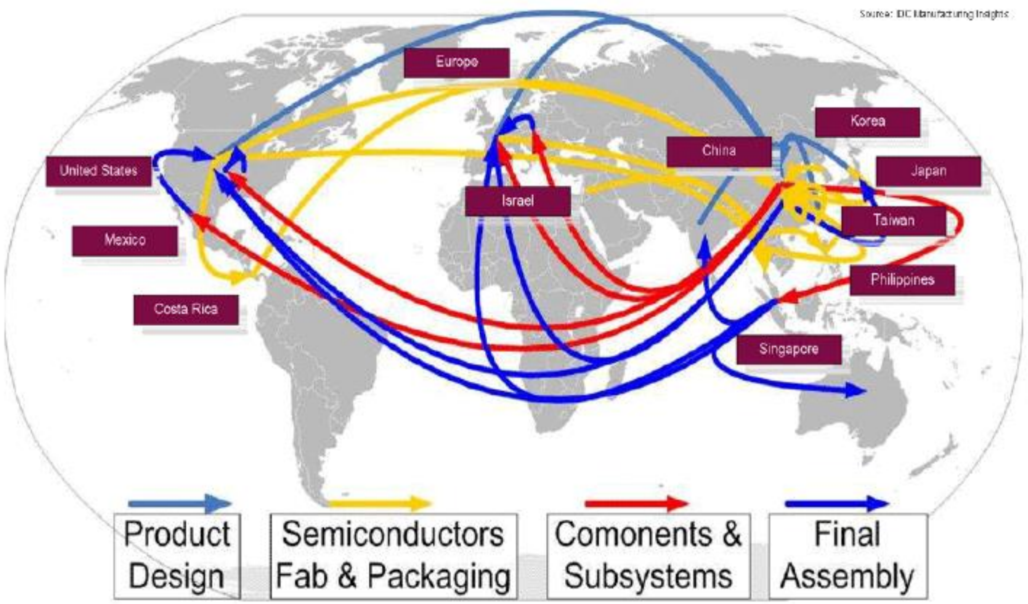
Yu Bi

ELE594 – Special Topic on Hardware Security & Trust

University of Rhode Island

Globalization of IC Supply Chain





- Economic concerns
- Time-to-market
- Design complexity

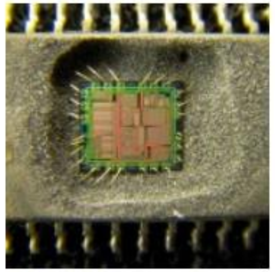
Security vulnerabilities

Security Vulnerabilities and Trust Issues

Counterfeiting

IP Piracy

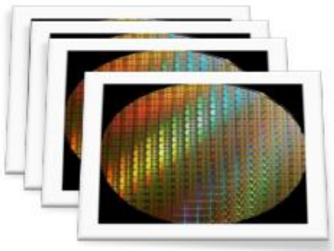
IC Piracy



Reverse Engineering

Hardware Trojans

Security Vulnerabilities and Trust Issues



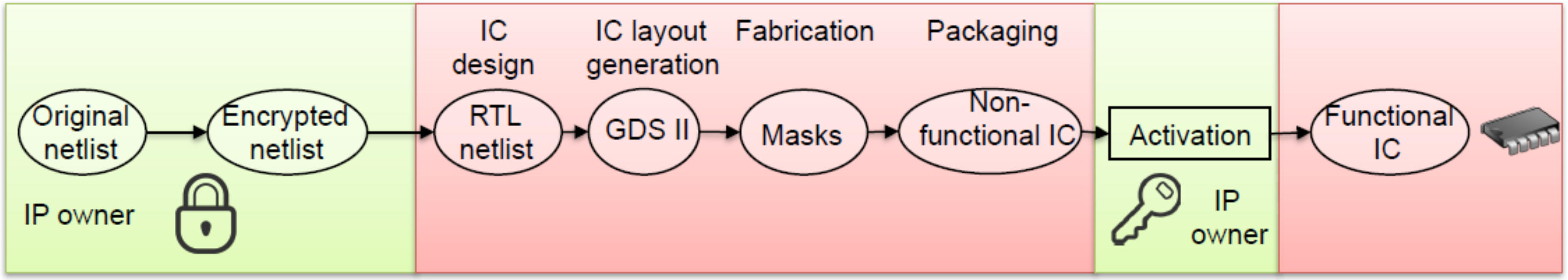
Impact

- Loss of revenue ~$4 billion annually
- Loss of trust
- Unreliable consumer electronics

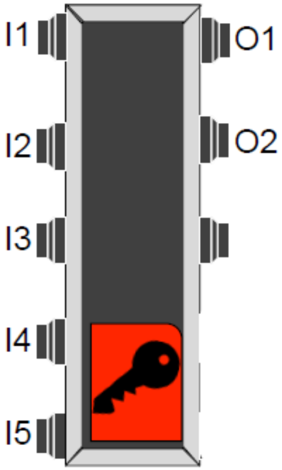
Reverse Engineering

Hardware Trojans

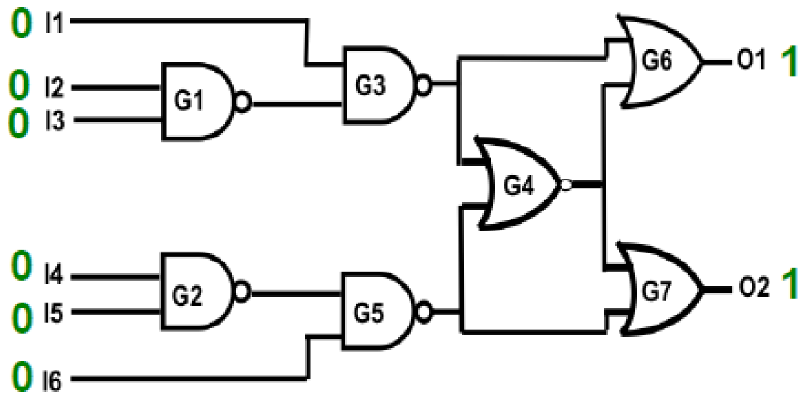
Logic Locking (LL)



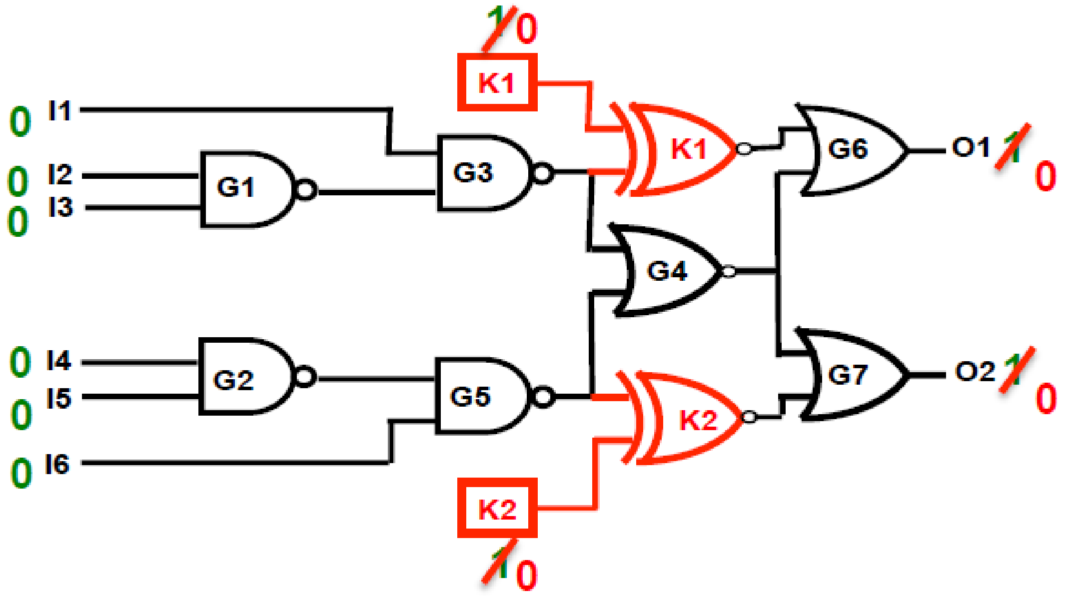
- Design for trust solutions:
  - Watermarking
  - Fingerprinting
  - IC metering
  - Logic encryption
- Logic locking/encryption/masking
  - IP owner encrypts/locks the netlist
  - IC is activated by loading the correct key



Logic Locking



Original netlist



Locked netlist

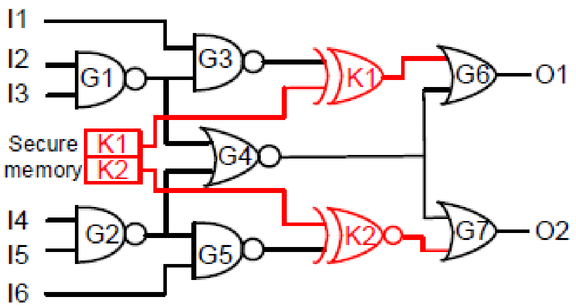
The circuit produces correct output only when the correct key is supplied.

Logic Locking Techniques

Random LL (RLL)[1]

Key-gates at random locations

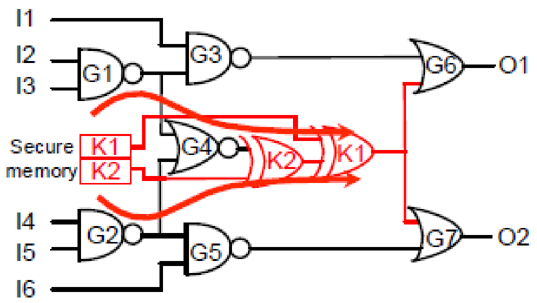
Key-gates uniformly distributed in the netlist



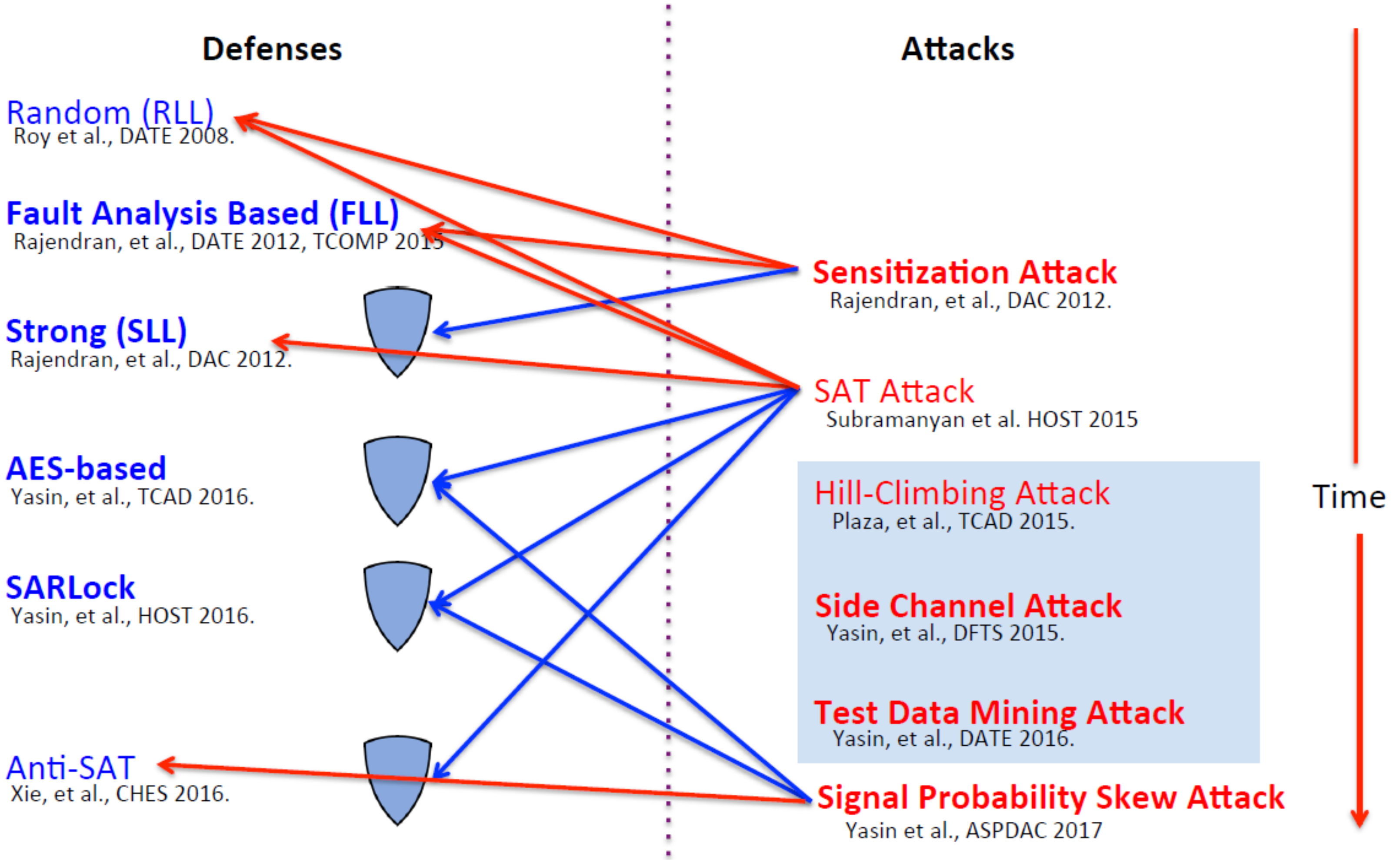
Fault analysis based LL (FLL)[2]

Key-gates at the most influential locations in the netlist

Key-gates tend to be localized and mostly back-to-back



Evolution of Logic Locking



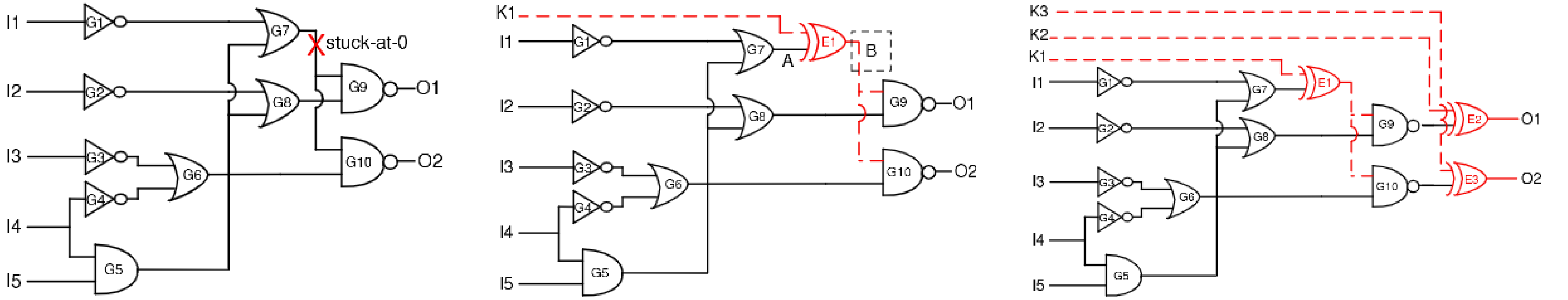
Attacks on Logic Locking

Sensitization attack	SAT attack	Signal probability skew attack
Threat model Locked netlist Functional IC	**Threat model** Locked netlist Functional IC	**Threat model** Locked netlist
Attack method Sensitize individual key bits to primary outputs	**Attack method** Eliminate incorrect keys using "distinguishing input patterns"	**Attack method** Trace the output of Anti-SAT block using signal skew as a trace
Defense Strong Logic Encryption	**Defense** SARLock, Anti-SAT	**Defense** SARLock

Attacks on Logic Locking

Sensitization attack	SAT attack	Signal probability skew attack
Threat model Locked netlist Functional IC	**Threat model** Encrypted netlist Functional IC	**Threat model** Encrypted netlist
Attack method Sensitize individual key bits to primary outputs	**Attack method** Eliminate incorrect keys using “distinguishing input patterns”	**Attack method** Trace the output of Anti-SAT block using signal skew as a trace
Defense Strong Logic Encryption	**Defense** SARLock, Anti-SAT	**Defense** SARLock

Fault Analysis-based LL



(a) A faulty circuit

(b) An encrypted circuit with a wrong key (K1 = 1) equivalent to the faulty circuit

(c) A circuit encrypted with three XOR gates (E1, E2, and E3)

Attack 1: Sensitization Attack

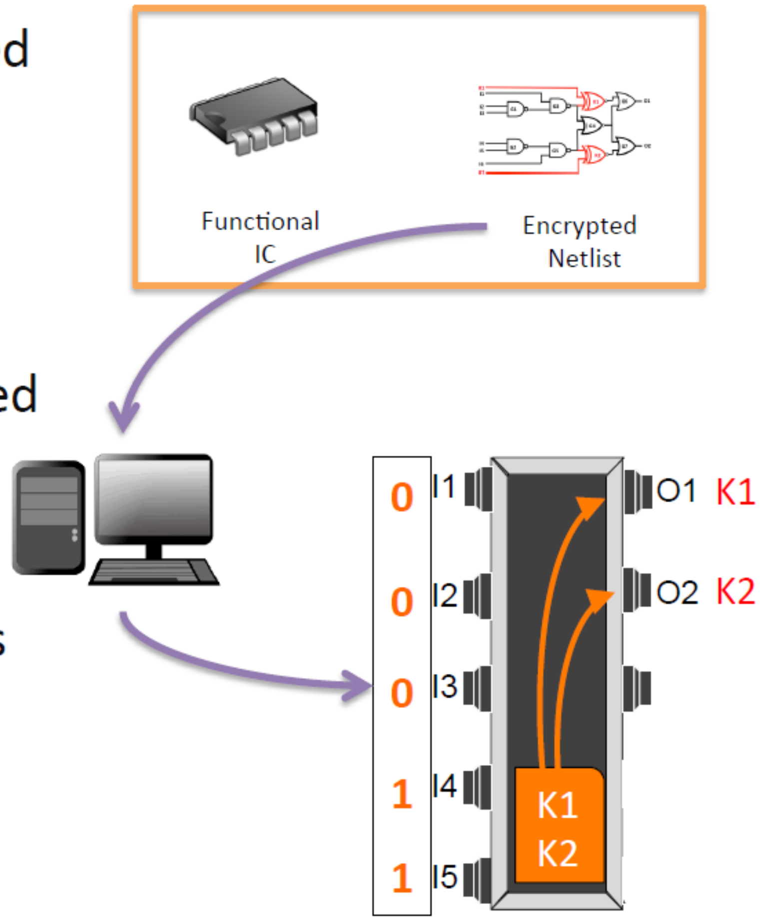
Goal: Determine the secret key used for logic encryption

Attacker has:

- Locked netlist
- Functional IC (with embedded key)

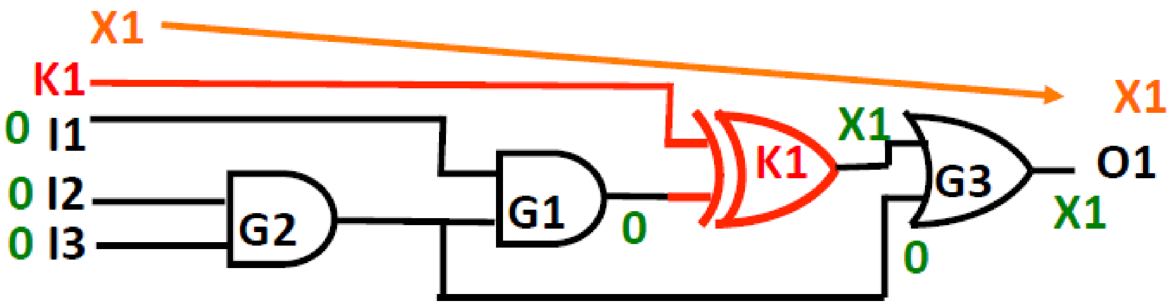
Attacker does:

- Compute the attack patterns from the locked netlist
- Applies them on IC
- Infers key from responses



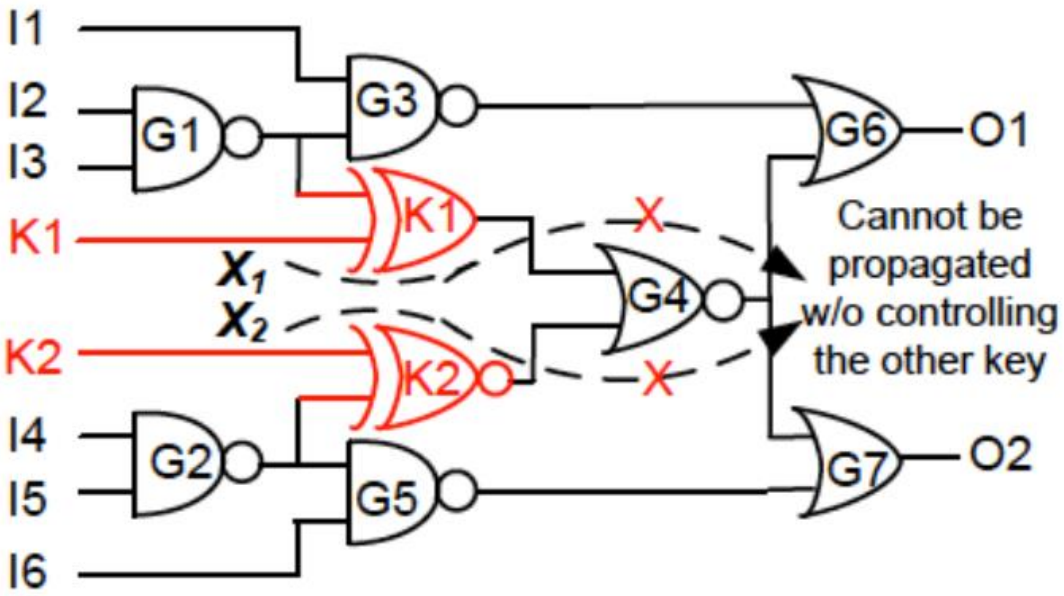
1. Rajendran, Jeyavijayan, et al. "Security analysis of logic obfuscation", DAC 2012

Sensitization Attack: Example



- **Objective**: Sensitize key K1 to primary output O1
- Find a test pattern to do sensitization
- Apply the test pattern to functional IC and observe the responses to find the value of key

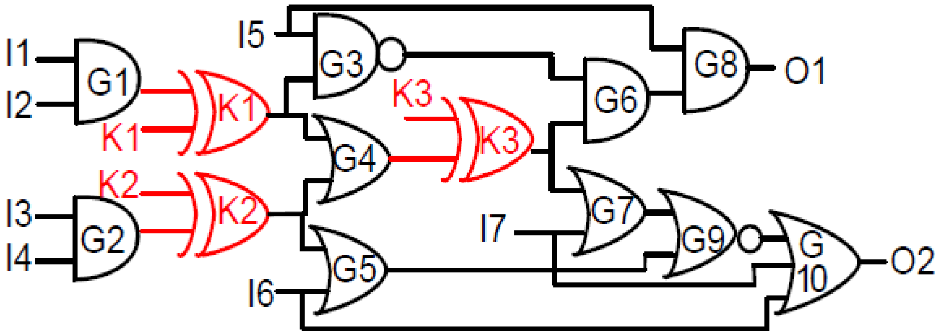
Solution: Strong Logic Locking (SLL)

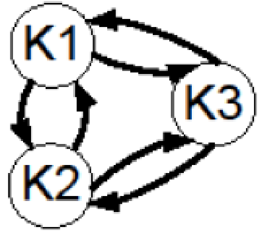


- Individual sensitization is not possible
- Pairwise secure key-gates
- Requires brute force:
  - Enumerate all possible values for the key bits
  - Exponential complexity!

SLL

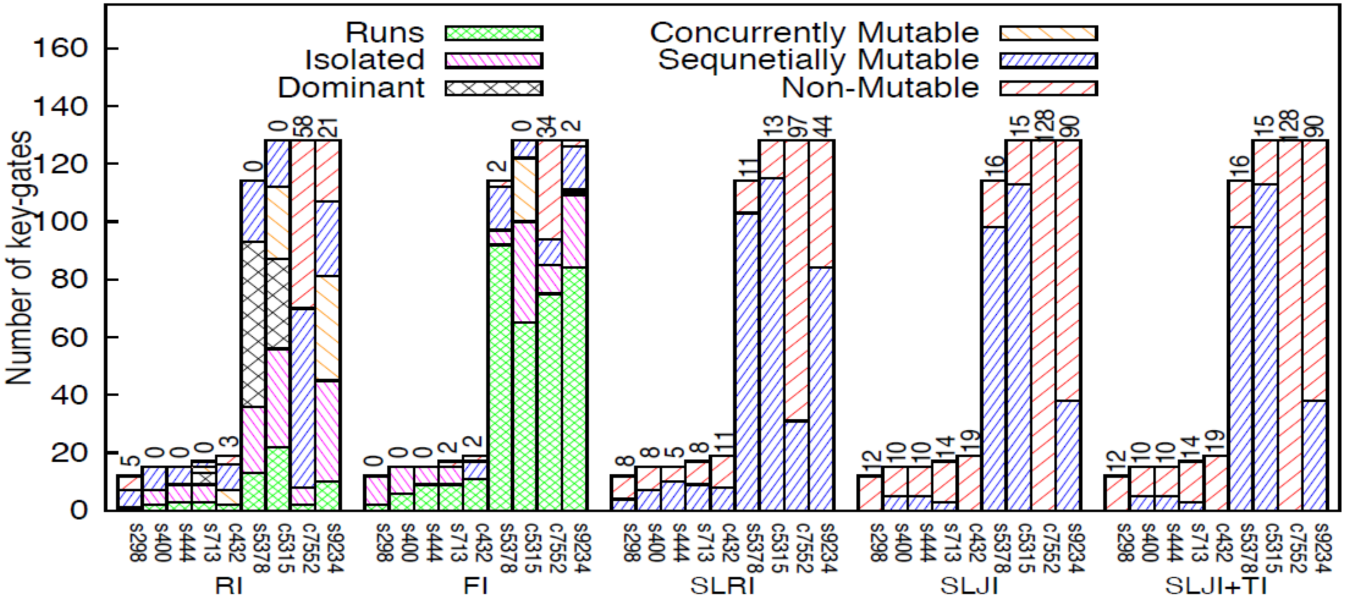
- **Interference graph**
  - Each gate is node and each edge has a type (e.g. mutable, non-mutable)
- **Security metric: Clique size**
  - Number of key-gates connected to one other by non-mutable edges





SLL Results: Clique Size

SLL Results: Clique Size



- SLJI achieves the largest clique size
  → **exponentially increasing effort** for the attacker

Attacks on Logic Locking

Sensitization attack

Threat model
Locked netlist
Functional IC

Attack method
Sensitize individual key bits to primary outputs

Defense
Strong Logic Encryption

SAT attack

Threat model
Locked netlist
Functional IC

Attack method
Eliminate incorrect keys using “distinguishing input patterns”

Defense
SARLock, Anti-SAT

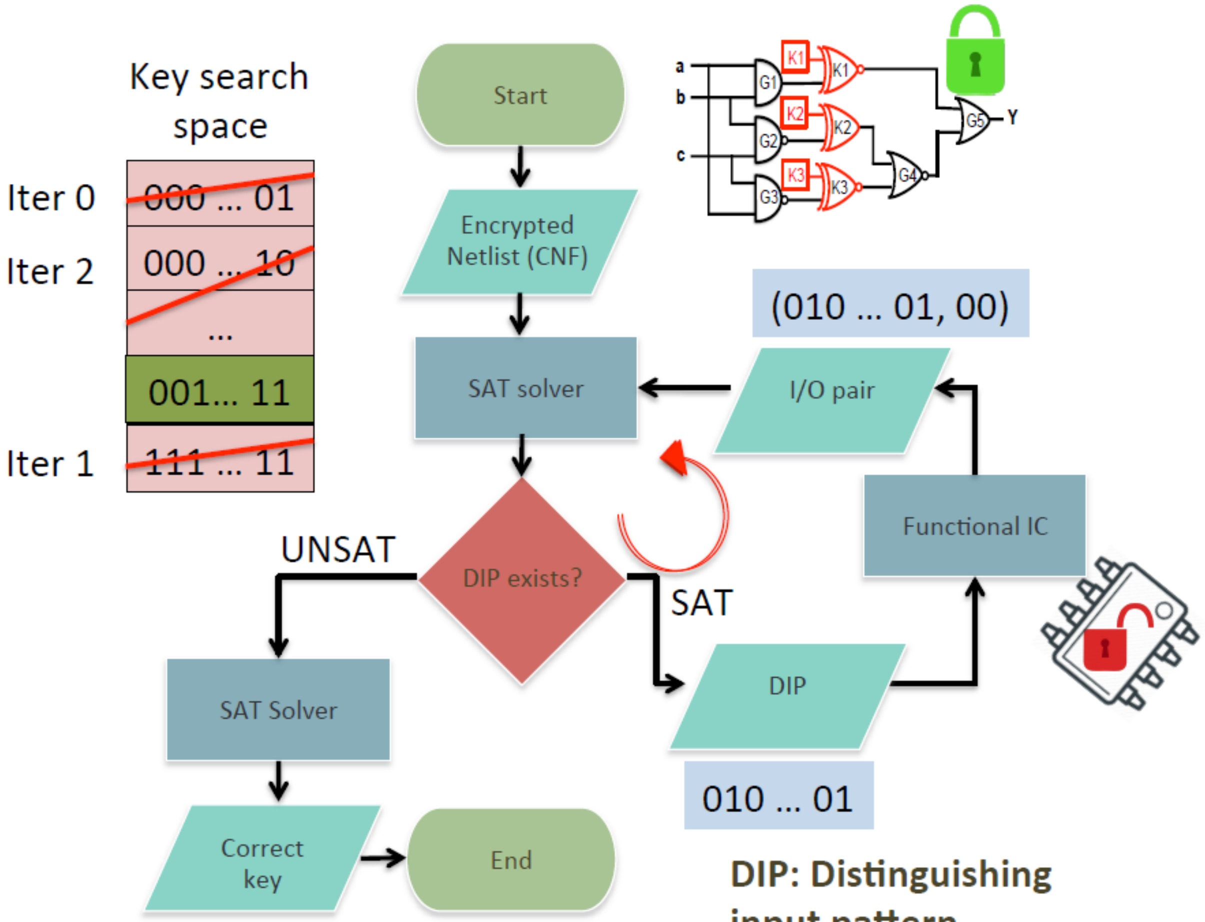
Signal probability skew attack

Threat model
Locked netlist

Attack method
Trace the output of Anti-SAT block using signal skew as a trace

Defense
SARLock

Attack 2: SAT Attack



DIP: Distinguishing input pattern

SAT Attack: Distinguishing Ability

					Output Y for different key values								
No.	a	b	c	Y	k0	k1	k2	k3	k4	k5	k6	k7	Pruned key values
0	0	0	0	0	1	1	1	1	1	1	0	1	
1	0	0	1	0	1	1	1	1	1	1	0	1	
2	0	1	0	0	1	1	1	1	1	1	0	1	
3	0	1	1	1	1	1	1	1	0	1	1	1	Iter 0: k4
4	1	0	0	0	1	1	1	1	1	1	0	1	Iter 3: all incorrect
5	1	0	1	1	1	1	1	1	1	1	1	0	Iter 2: k7
6	1	1	0	1	1	1	0	1	1	1	1	1	
7	1	1	1	1	1	0	1	1	1	1	1	1	Iter 1: k1

- Each DIP eliminates a different # of key values
- Keys pruned by a DIP↑ →Computational complexity ↓

SAT Attack: Experimental Results

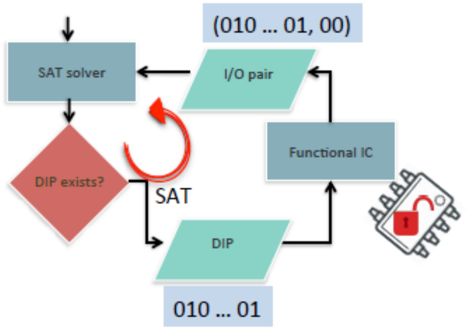
- Strong logic locking (SLL)
  - Broken using a small # of DIPs

Benchmark	#DIPs					Execution Time (s)				
	11	12	13	14	15	11	12	13	14	15
s5378	8	9	9	10	13	0.2	0.2	0.2	0.2	0.2
c5315	4	3	4	5	3	0.3	0.3	0.3	0.3	0.3
c7552	8	9	9	9	12	0.7	0.5	0.5	0.5	0.5
s9234	7	13	13	10	12	0.2	0.3	0.3	0.3	0.3
IFU	8	8	9	13	11	0.1	0.1	0.1	0.1	0.1
LSUrw	4	5	5	7	9	0.1	0.1	0.1	0.1	0.1
FPUin	6	7	8	5	9	0.1	0.1	0.1	0.1	0.1
LSUex	5	5	8	8	6	0.1	0.1	0.1	0.1	0.1
SB	7	5	6	6	6	0.1	0.1	0.1	0.1	0.1
IFQ	9	7	9	9	8	0.2	0.2	0.2	0.2	0.2
TLU	7	6	7	9	10	0.3	0.3	0.4	0.3	0.4

SLL is vulnerable to the SAT attack !!

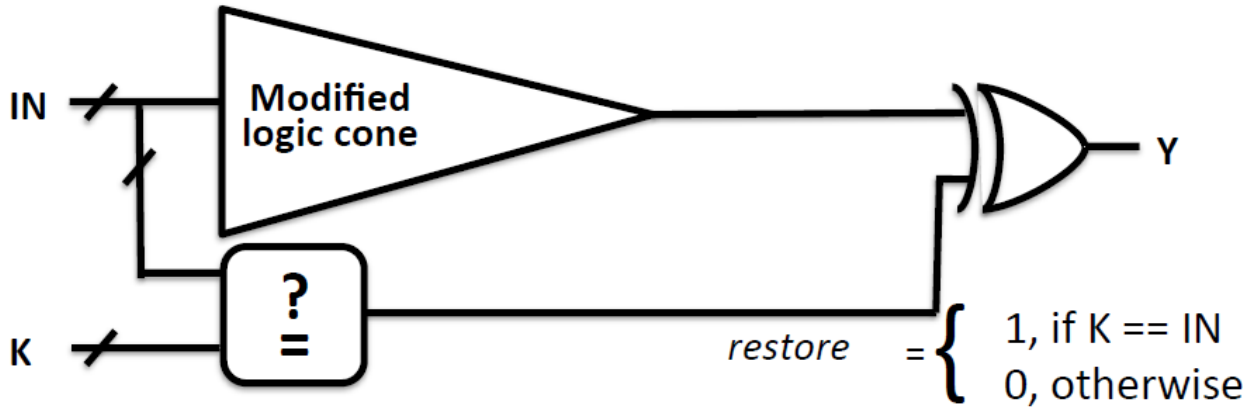
Thwarting SAT Attack

					Output Y for different key values							
No.	a	b	c	Y	k0	k1	k2	k3	k4	k5	k6	k7
0	0	0	0	0	0	1	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	1	0	0	0	0
3	0	1	1	1	1	1	1	1	1	1	1	1
4	1	0	0	0	0	0	0	0	0	1	0	0
5	1	0	1	1	1	1	1	1	0	1	1	1
6	1	1	0	1	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1	1	1	1	0



- Desired: Each DIP eliminates one key value
- # of DIPs = Number of input combinations

Solution 1: SARLock



SAT attack resistant LL

- Original logic cone is modified for one input pattern
- The modification is restored using the comparator block

					Output Y for different key values							
No.	a	b	c	Y	k0	k1	k2	k3	k4	k5	k6	k7
0	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	1	0	0	1	0	0	0	0	0	0
2	0	1	0	0	0	0	1	0	0	0	0	0
3	0	1	1	1	1	1	1	0	1	1	1	1
4	1	0	0	0	0	0	0	0	1	0	0	0
5	1	0	1	1	1	1	1	1	1	0	1	1
6	1	1	0	1	1	1	1	1	1	1	0	1
7	1	1	1	1	1	1	1	1	1	1	1	0

# of DIPs = 2^{k-1}

SARLock: Experimental Results

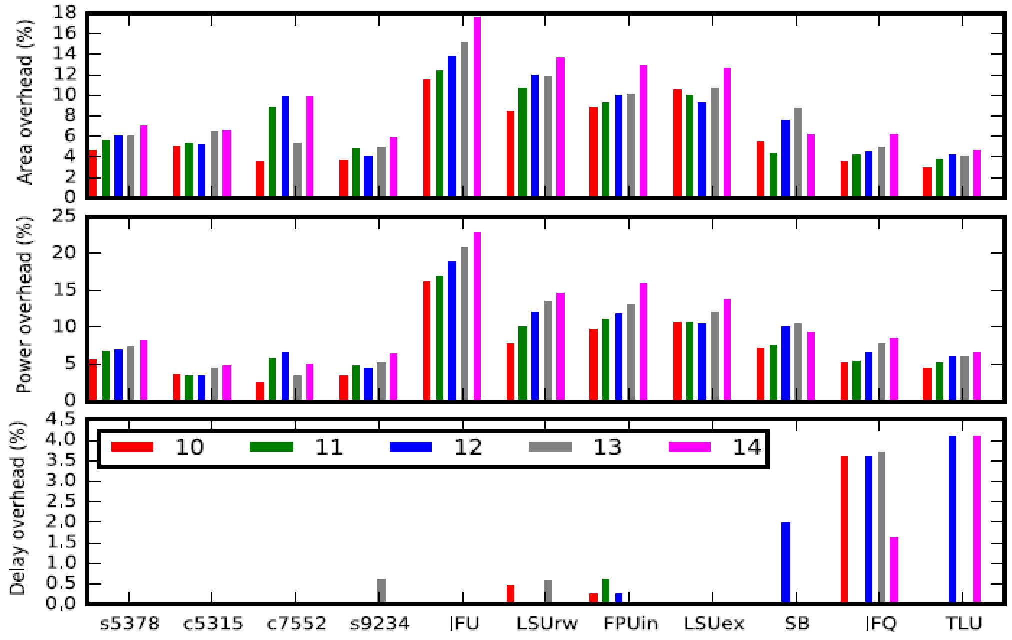
- SARLock
  - #DIPs ≈ $2^{|K|-1}$
  - Key size ↑ ➔ Execution time (3x-4x) ↑

Benchmark	#DIPs					Execution Time (s)				
	11	12	13	14	15	11	12	13	14	15
s5378	1024	2048	4096	8191	16384	54.1	190.6	619.7	4351.8	10250.7
c5315	1024	2049	4096	8191	16383	75.4	252.9	829.1	4778.2	15874.9
c7552	1025	2049	4096	8191	16386	78.3	234.1	757	3165.3	14573.1
s9234	1027	2049	4102	8195	16386	77.2	247.9	864.1	3225.7	15532.3
IFU	1023	2056	4100	8206	16389	55.2	166.7	789.5	2309.8	10258.7
LSUrw	1025	2049	4096	8194	16383	58.2	152	626.9	1802.6	7466.6
FPUin	1025	2049	4097	8194	16384	28.4	135	1359.6	4497.6	15457.2
LSUex	1024	2049	4096	8194	16384	52.8	268.3	1137.2	3101.3	16707.1
SB	1026	2050	4099	8194	16386	69.2	257.4	1416.6	3304.6	19193.7
IFQ	1024	2048	4098	8192	16384	63.3	290.8	1644.7	4185.4	14563.1
TLU	1027	2052	4099	8195	16385	57.2	227	2238.7	3507.6	18760.3

SARLock resists SAT attack

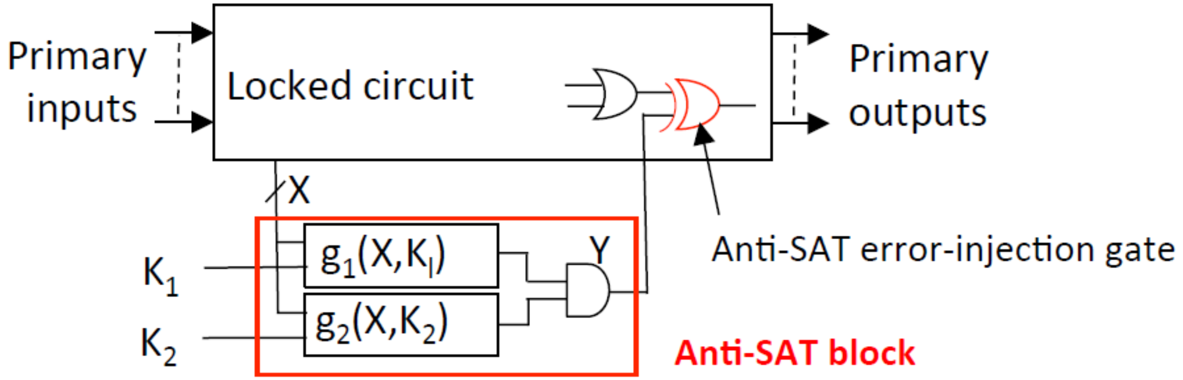
SARLock: Experimental Results

- SARLock
  - Exponential security gain at linear increase in cost



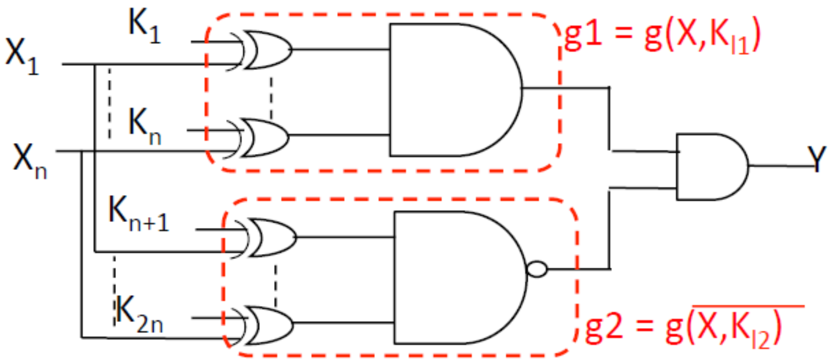
Minimal delay overhead

Solution 2: Anti-SAT



- Designed to integrate with a locked circuit
- Consists of two complementary functions
  - Control/reduce number of keys eliminated by a DIP
- In the best case
  - one key eliminated by each DIP ➔ #DIPs exponential
  - AND/NAND, OR/NOR gates used to construct Anti-SAT block

Anti-SAT: SAT Attack Resilience



Correct key

g1 and g2 are complementary

Y=0 for all input values

No error injected

Locked func. = original func.

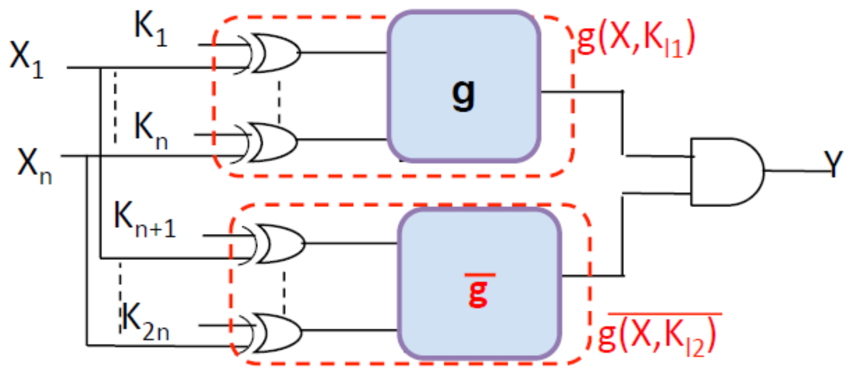
Incorrect key

g1 and g2 complementary for all input values except one

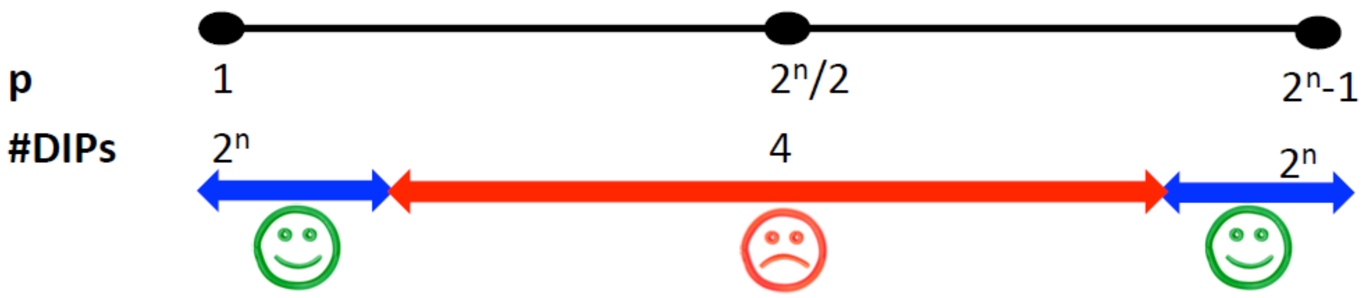
Y=1 (error injected) for exactly one key for any input value

SAT attack effort: #DIPS = 2^n -1

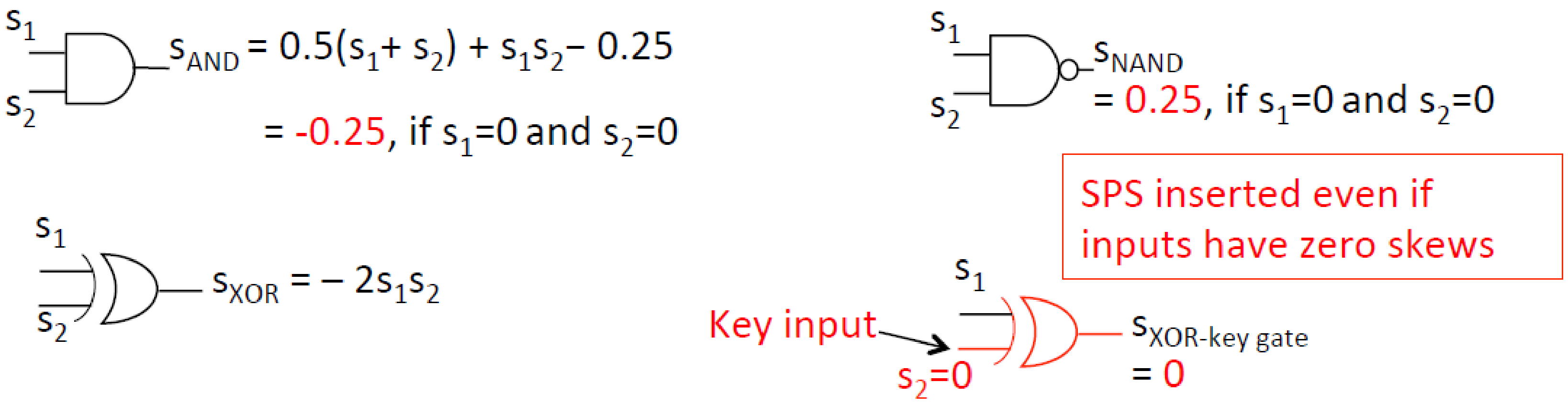
Anti-SAT: Generic Functions



- For an AND gate, |on-set| = 1
- For a generic g, |on-set| = p

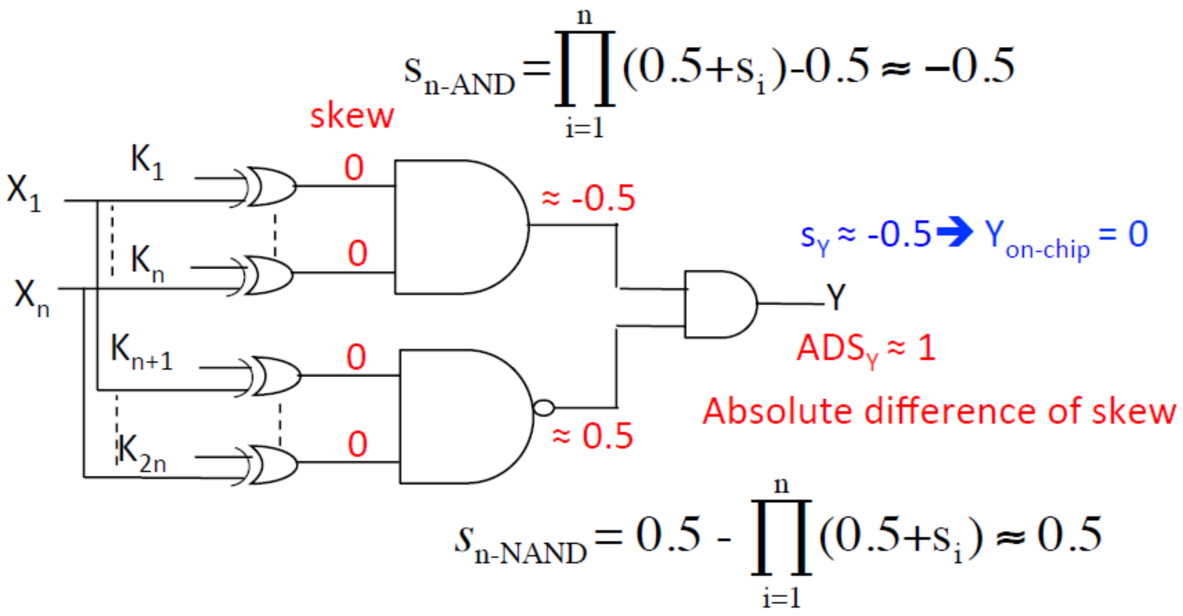


Attack 3: SPS Attack



- Anti-SAT construction → structural traces
- Signals skewed/biased towards either 0 or 1
- Output gate Y has inputs skewed oppositely
- Signal probability skew, s = Pr [x=1] - 0.5
  - For a primary input, Pr [in=1]=0.5 , sin = 0

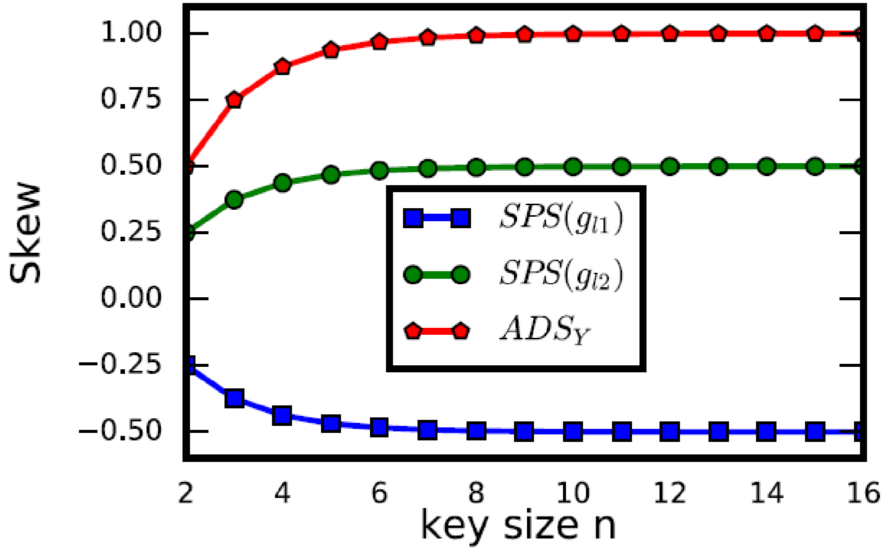
SPS Attack: Operation



- ADS_Y serves as a trace for identifying Anti-SAT block
- s_Y determines the correct value of Y
- Gates with such high skewed ADS values are rare
- As key size n increases, ADS_Y is closer to 1

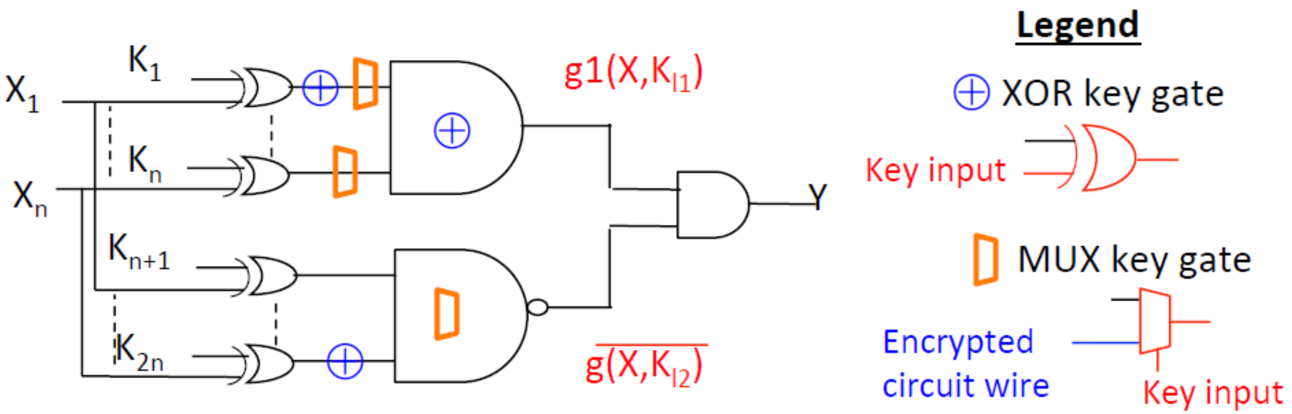
SPS Attack Results: Impact of Key Size

- Impact of key size (n) on ADSY for basic Anti-SAT (BA)
  - With increasing n, the ADSY ≈ 1



Key size ↑ → attack effectiveness ↑

Functional Obfuscation



Functional obfuscation

Breaks symmetry of signals in the Anti-SAT block

Inserts n XOR key-gates

One input is a wire in Anti-SAT block, other is a key input

Structural obfuscation

Hides whether a signal belongs to Anti-SAT or locked circuit

Inserts n MUXes

One input is a wire in Anti-SAT block, other is a wire in the encrypted circuit

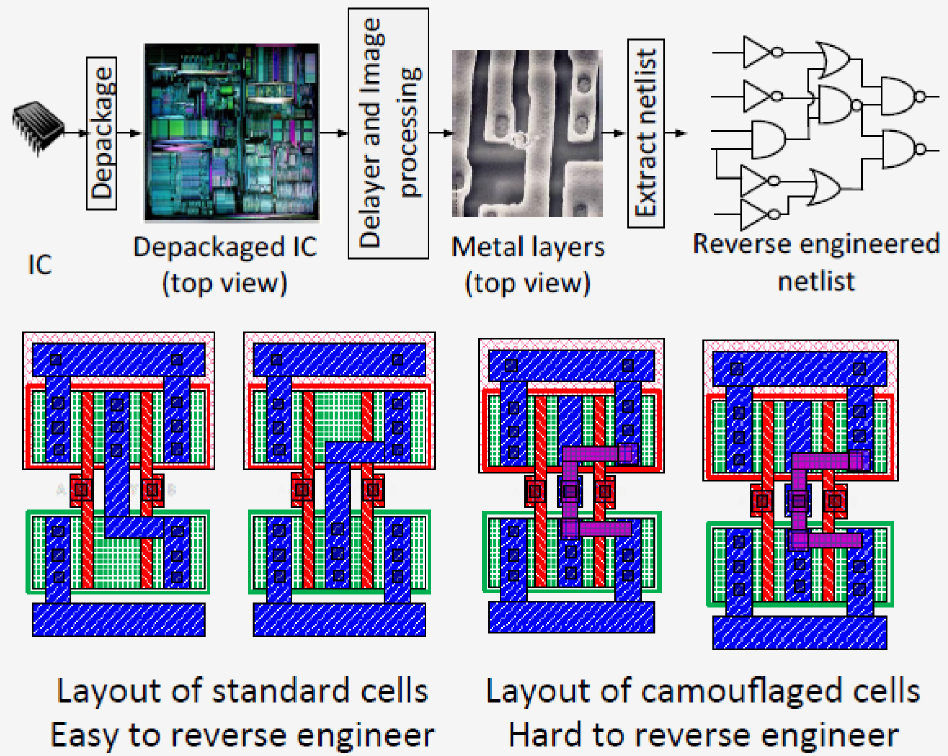
SPS Attack Results: Obfuscation

- FLL(5%)+64-bit obfuscated Anti-SAT
  - Anti-SAT inputs = Random wires in FLL-locked circuit
  - Random integration using MUXes changes ADS_Y slightly

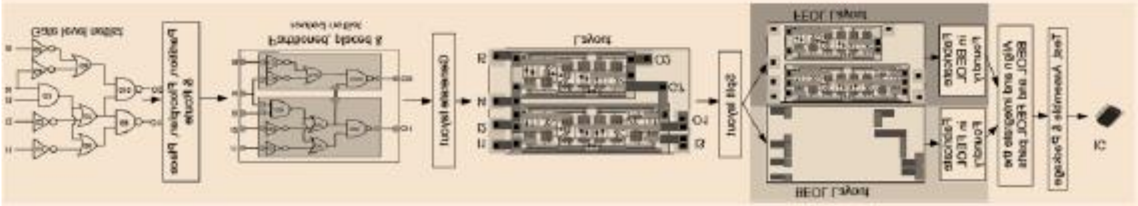
Benchmark	ADS_Y	#cand.	Exec. time (sec)
fpu_div	0.999973	1	1
lsu_stb	0.999973	1	1
c5315	0.999969	1	1
c7552	0.999973	1	2
ifu_ifq	0.999969	1	2
tlu_mmu	0.999971	1	2
s13207	0.999973	1	7
s15850	0.999971	2	22
s38584	0.999972	1	83
s38417	0.999973	1	93

SPS attack is effective against obfuscated Anti-SAT

Other IP Protection



(Anti-)Reverse Engineering Techniques



Split manufacturing for IP protection