A Conflict-Free Approach for Parallelizing SAT-Based De-Camouflaging Attacks

Xueyan Wang[1], Qiang Zhou[1], Yici Cai[1], Gang Qu[2]

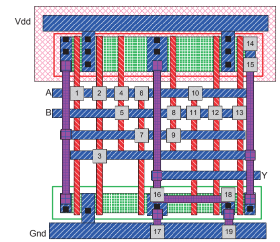
1. Tsinghua University
2. University of Maryland, College Park

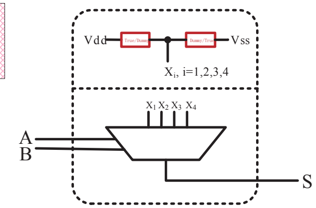
Outline

- Background
- Contributions
- Two-Level Circuit Partitioning
- Conflict-Free Parallelization Framework
- Experimental Results
- Summary

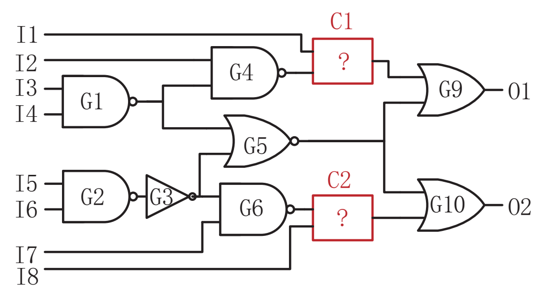
Gate Camouflaging Against RE

- Gate Camouflaging
  - Selective gates are replaced by camouflaged cells
  - Camouflaged cells appear identical look with different functionalities
- De-camouflaging Attacks
  - Brute force attack
  - IC testing based attack
  - Circuit partition attack
  - SAT-based attack



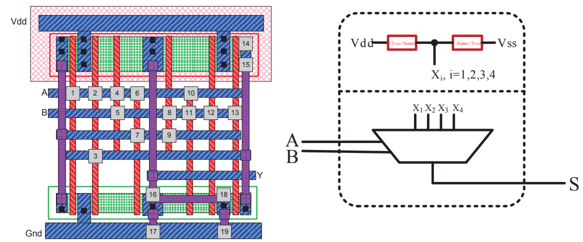


Various camouflaged cells

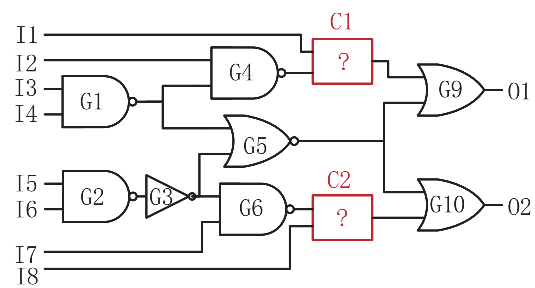


Gate Camouflaging Against RE

- Gate Camouflaging
  - Selective gates are replaced by camouflaged cells
  - Camouflaged cells appear identical look with different functionalities
- De-camouflaging Attacks
  - Brute force attack
  - IC testing based attack
  - Circuit partition attack
  - SAT-based attack

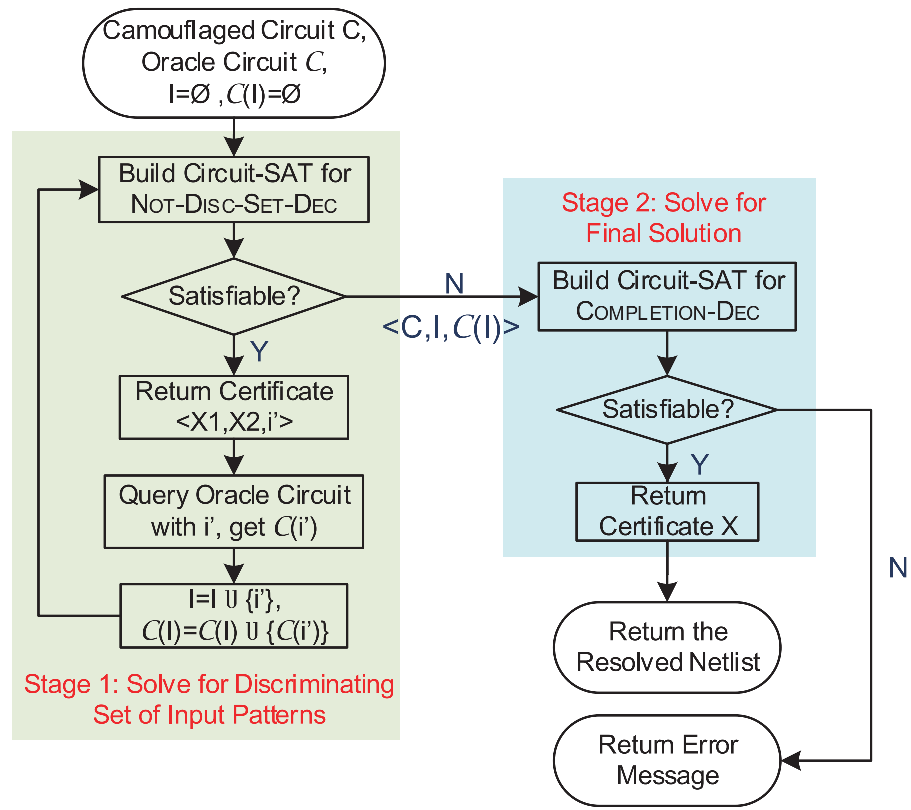


Various camouflaged cells



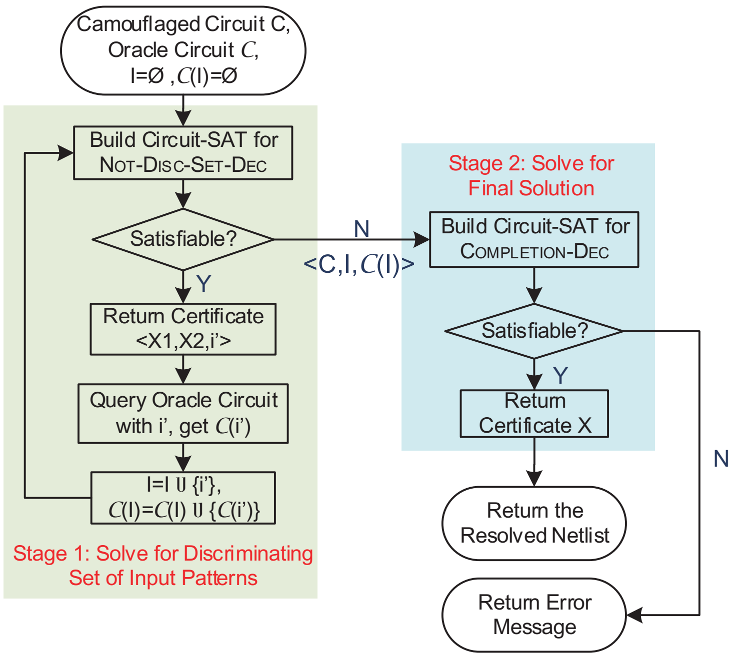
What is SAT-Based De-Camo Attack?

- Key idea: Prune all incorrect assignments with a discriminating set of input patterns (DiscSet).
- Method:
  - Stage 1: Find DiscSet: Iteratively find new input pattern, until the set is discriminating.
  - Stage 2: Find the correct assignment with DiscSet.



Why Need Parallelization?

- X= *|DiscSET|*
- Need to call SAT solvers by *(X+2)* times
  - Stage 1 finding DiscSET need to call SAT solver (x+1) times;
  - Stage 2 finding correct assignment need to call SAT solver for one time;



- With #Camo-gates or circuit size grows
  - #variables and #clauses in SAT formulas increase
  - #calling for SAT solver increases
  - SAT-based attacks become less effective

Outline

- Background
- Contributions
- Two-Level Circuit Partitioning
- Conflict Avoidance Strategy
- Parallelization Framework
- Experimental Results
- Summary

Contributions

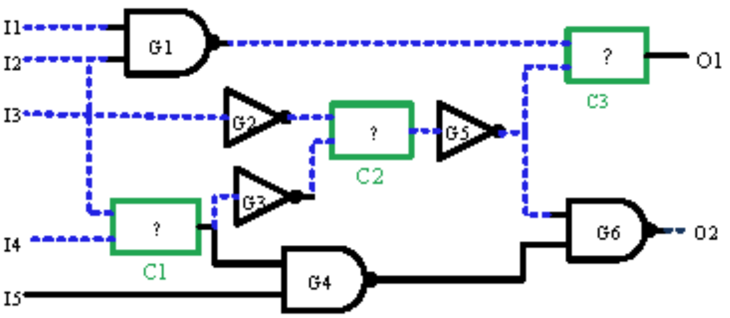
- Dividing SAT-based attack into smaller sub-problems
  - Independent module partitioning
  - K-medoids clustering
- Avoiding conflicts while solving sub-problems
- A parallelization framework for SAT-based de-camouflaging attacks

Outline

- Background
- Contributions
- Two-Level Circuit Partitioning
- Conflict-Free Parallelization Framework
- Experimental Results
- Summary

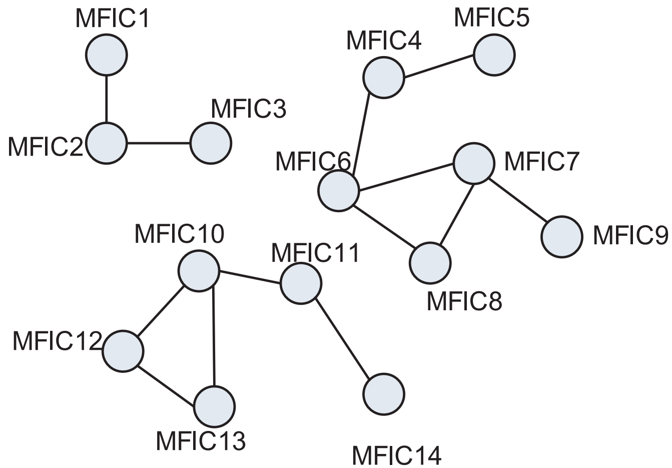
How to do Partitioning?

- Definition of MFIC
  - $MFIC_{PO}$={G | Exists path, G→PO}

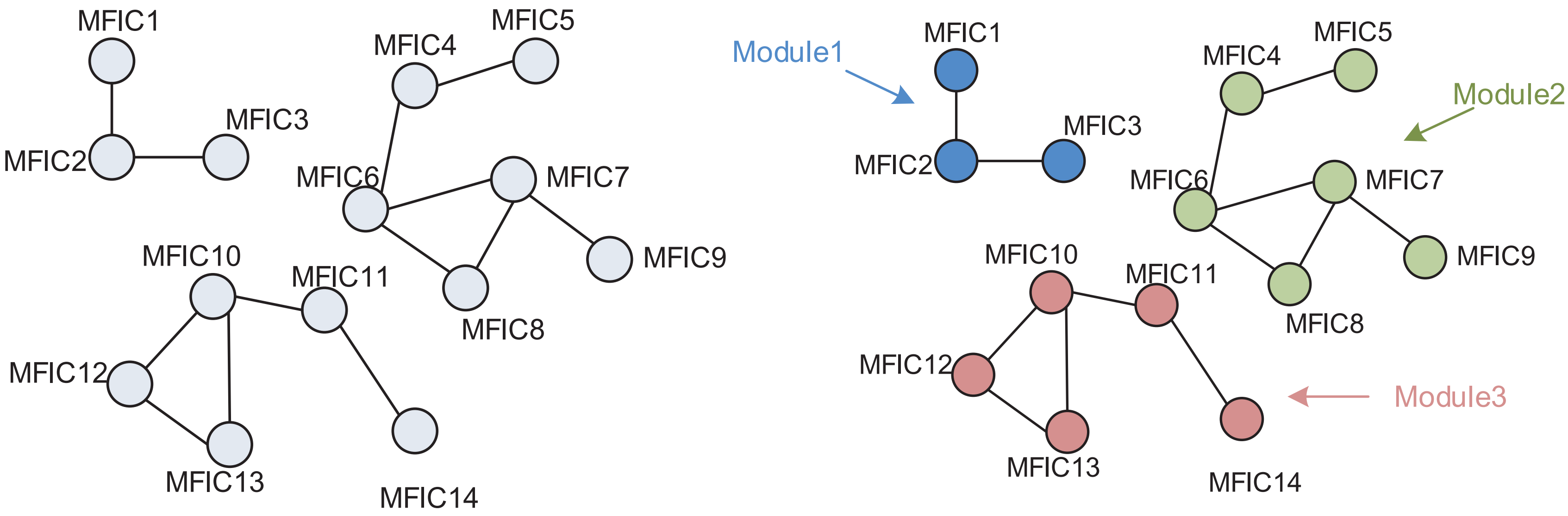


- Construct a graph G

$$V = \{MFIC_i | \exists CG \in MFIC_i\},$$
$$E = \{(v_i, v_j) | \exists CG \in (v_i \cap v_j); v_i, v_j \in V, i \neq j\}.$$

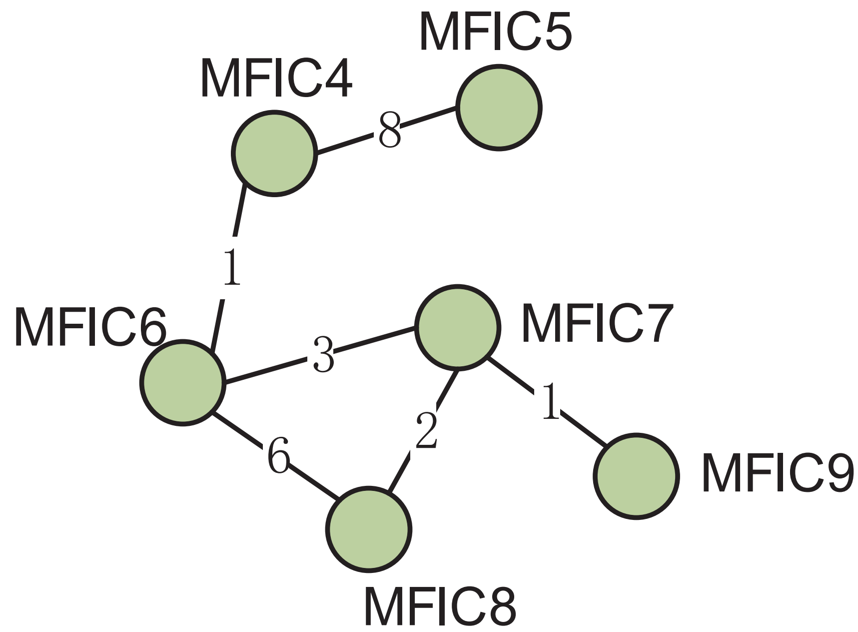


Independent Module (IM) Partitioning



- IMs do not share camouflaged gates
- Module 1, Module 2, and Module 3 are independent, can be de-camouflaged independently

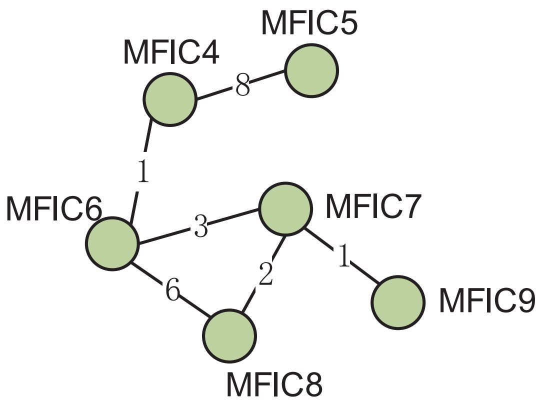
Further Partitioning Within IMs

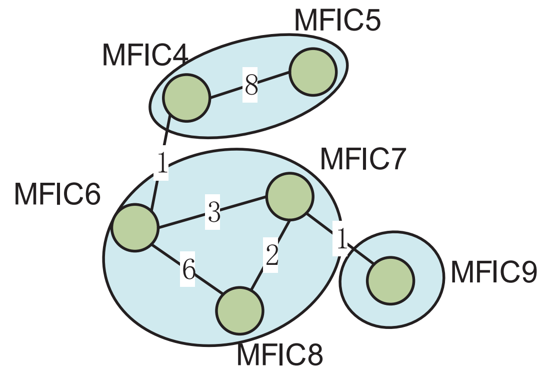


- Treat the IM as a whole? Exists ultra large modules!
- Totally partitioning? Too many repetitive efforts!

K-Medoids Clustering

- Balance Scale Reductions VS. Repetitive Efforts





- Perform k-medoids clustering
  - Define Weight: #Common Camo Gates
  - Number of clusters k:

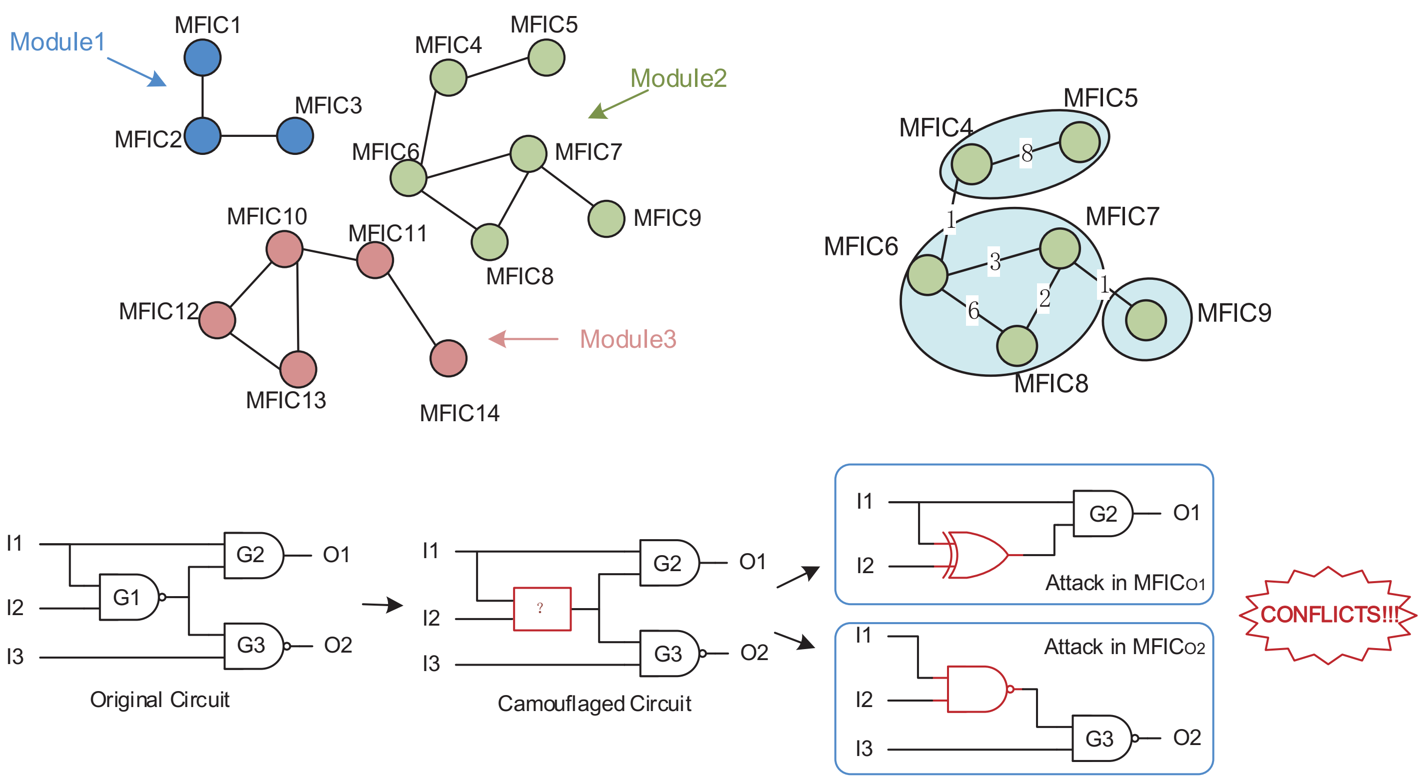
$$k = \left\lceil \frac{|MFIC_{IM}| * |CG_{IM}|}{2 * \sum_i CG_{MFIC_i}} \right\rceil \qquad \frac{|CG_{IM}|}{\sum_i CG_{MFIC_i}} \in (0, 1).$$

Outline

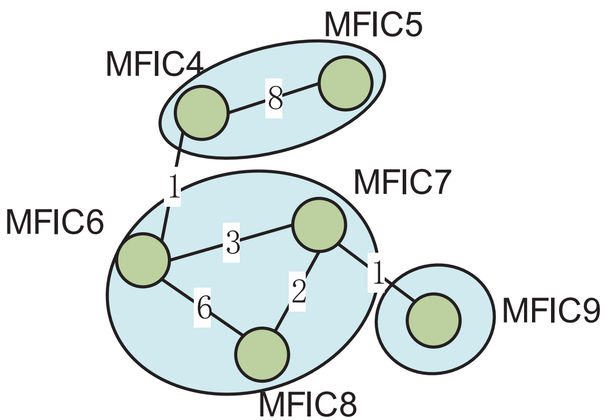
- Background
- Contributions
- Two-Level Circuit Partitioning
- Conflict-Free Parallelization Framework
- Experimental Results
- Summary

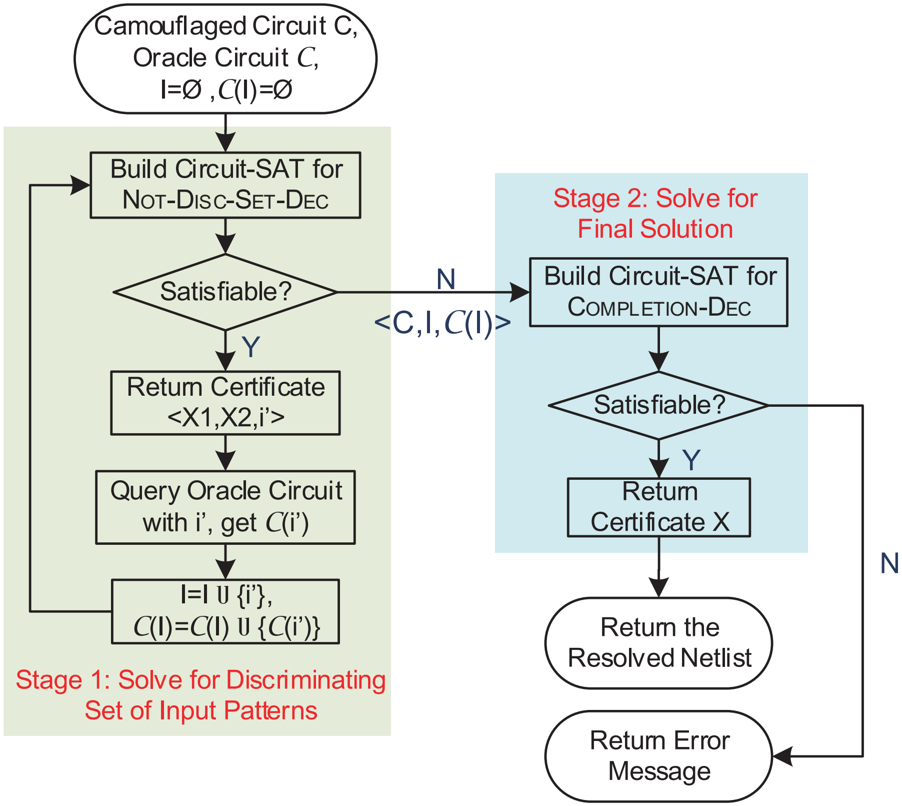
How Conflicts Happen?

- Conflict: a camouflaged gate is designated with different functionalities while being attacked in different sub-circuits.



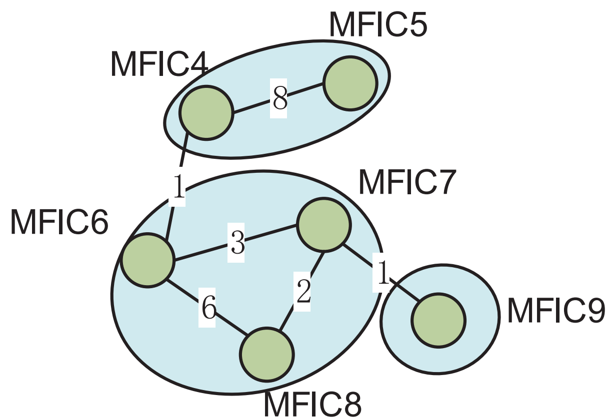
How to Avoid Conflicts?

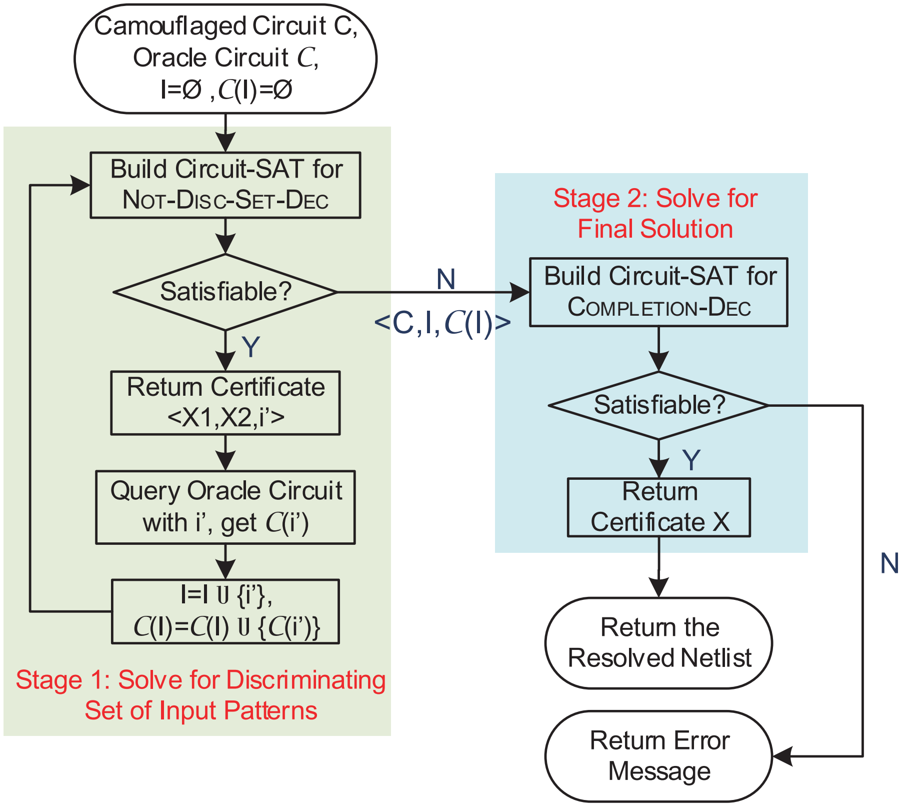




- Stage 2 designate camouflaged gates with certain functionalities
- Only perform Stage 1 in clusters, then perform Stage 2 within the whole module.

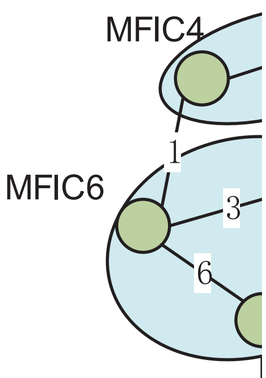
How to Avoid Conflicts?





- *Theorem 1: The union set of* DiscSets *of clusters in one IM, is the IM's one* DiscSet.

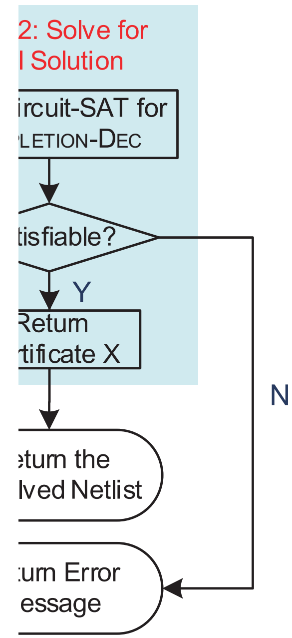
How to Avoid Conflicts?



Theorem 1. *The union set of DiscSets of clusters in one IM, is the IM's one DiscSet. Namely* $IM = \bigcup_i Cluster_i \Rightarrow$

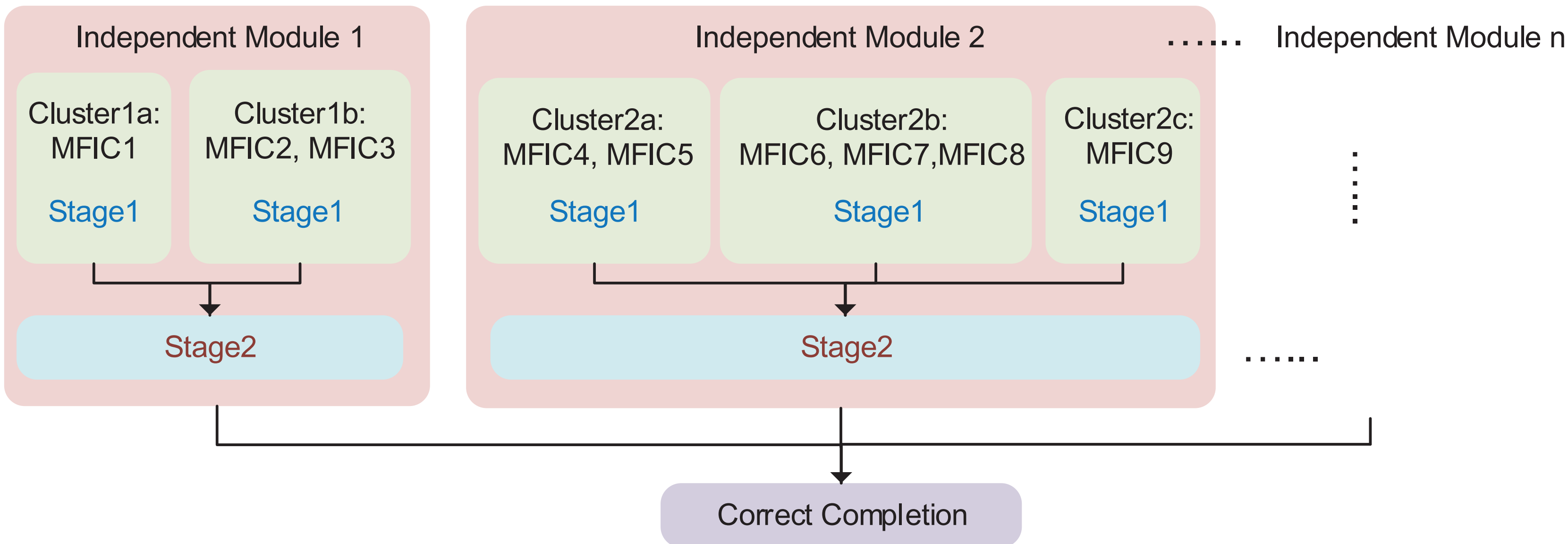
$$DiscSet_{IM} = \bigcup_i DiscSet_{Cluster_i}.$$

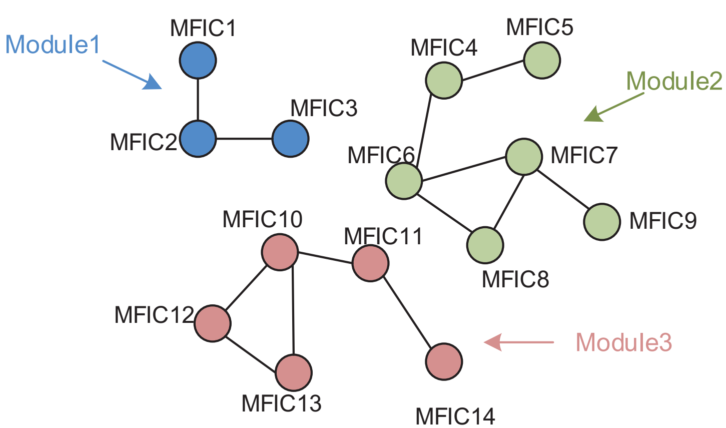
Proof. In each $Cluster_i$, any completion under $DiscSet_{Cluster_i}$ can guarantee that the resolved cluster have the same function as the corresponding oracle circuit. Therefore, under $\bigcup_i DiscSet_{Cluster_i}$, all the clusters will have the same function with corresponding sub-circuits in the oracle circuit. Given that $IM = \bigcup_i Cluster_i$, therefore, IM will have the same function with the corresponding oracle circuit. As a result, $\bigcup_i DiscSet_{Cluster_i}$ is one discriminating set of input patterns for IM. □

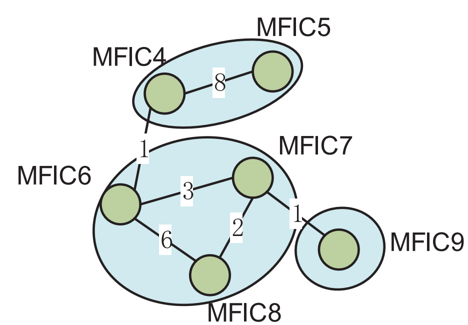


- *Theorem 1: The union set of* DiscSets *of clusters in one IM, is the IM's one* DiscSet.

Conflict-Free Parallelization Framework







Outline

- Background
- Contributions
- Two-Level Circuit Partitioning
- Conflict-Free Parallelization Framework
- Experimental Results
- Summary

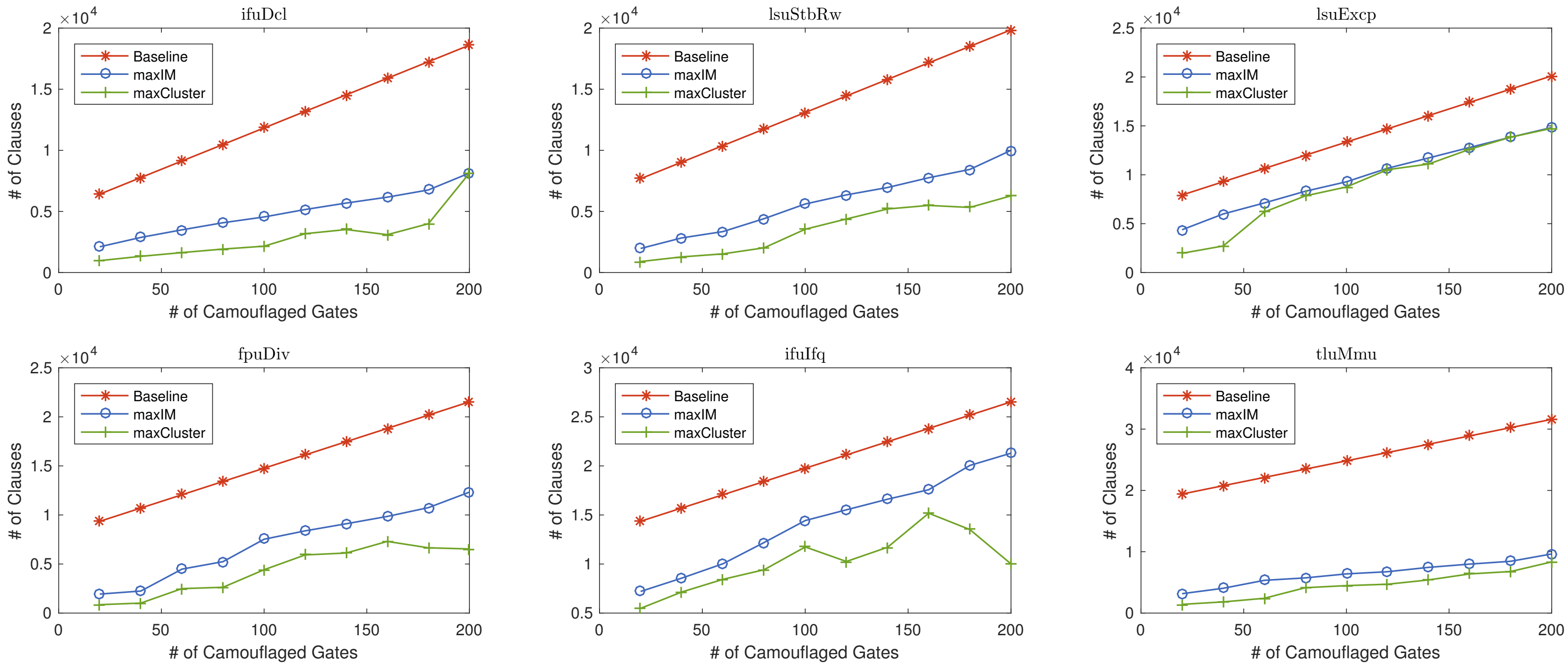
Scale Reductions of SAT Formulas

Sat-Based De-Camouflaging Attack: Our Approach Vs. Baseline Approach When 20 Gates are Camouflaged in ifuDcl .

IM Index	CLAUS	VARS	ITERS	CPU Time (s)
1	1805	804	19	0.156286
2	1676	731	23	0.156286
3	145	68	5	0.003147
4	155	72	5	0.003147
5	137	64	4	0.002403
6	145	68	5	0.002815
7	213	100	6	0.005208
8	145	68	5	0.003624
9	113	54	5	0.008026
10	155	72	5	0.003434
-	-	-	-	1.86755
Baseline	6421	2696	51	4.05547

- #CLAUS, #VARS, #ITERS achieves **71.9%**, **70.2%**, and **55%** reductions, respectively.
- Total runtime reduces from **4.1s** to **1.9s**.

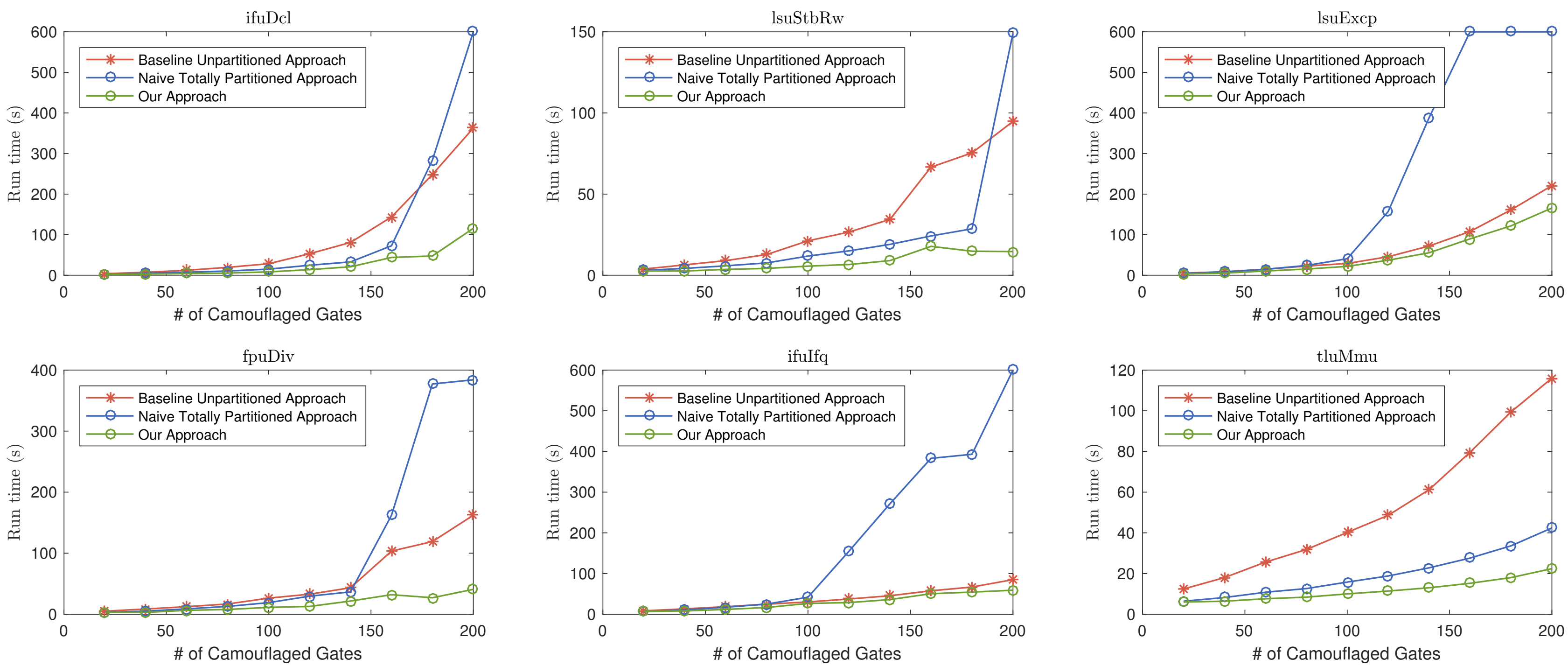
Scale Reductions of SAT Formulas



Comparisons of #CLAUS in SAT formulas.

- #CLAUS in the largest IMs and the largest clusters are on average reduced by 53% and 67%, respectively.
- Increase much slower than the baseline approach with more gates being camouflaged.

Speed Up of Parallelized SAT-Based Attack



Comparisons for the runtime by our approach, baseline unpartitioned approach, and naive totally partitioned approach.

- Achieves an average of 3.6x and up to 10x speed up than baseline.
- Naive totally partitioned method makes the attack slower.

Outline

- Background
- Contributions
- Two-Level Circuit Partitioning
- Conflict-Free Parallelization Framework
- Experimental Results
- Summary

Summary

- A conflict-free method to parallelize SAT-based de-camouflaging attacks is proposed.
  - Independent module partitioning
  - k-medoids clustering
  - conflict avoidance strategy
- Experiments demonstrate on average 50% scale reduction and 3.6x speed up over the state-of-the-art fastest de-camouflaging tool.

Thank you!