FANCI: Identifying Malicious Circuits

Adam Waksman
Matthew Suozzo
Simha Sethumadhavan

Computer Architecture & Security Technologies Lab
Department of Computer Science
Columbia University
Do You Trust Hardware?

Cyber-attack concerns raised over Boeing 787 chip's 'back door'
Researchers claim chip used in military systems and civilian aircraft has built-in function that could let in hackers

NSA Subverts Most Encryption, Works With Tech Organizations For Back-Door Access, Report Says

Western spooks banned Lenovo PCs after finding back doors

Report suggests 'Five Eyes' alliance won't work with Chinese PCs

NSA’s Own Hardware Backdoors May Still Be a “Problem from Hell”

The Problem of Third-Party IP

Increase in Usage of Third-Party IP in Phones

![Graph showing the increase in usage of third-party IP in phones](International Business Strategies, 2012)
Our Solution

- Automatically identify malicious circuits in third-party hardware design IP.

```verilog
assign bus_x87_i = arg0 & arg1;
always @(posedge clk) begin
  if (rst) data_store_reg7 <= 16'b0;
  else begin
    if (argcarry_i37 == 16'hbacd0013) begin
      data_store_reg7 <= 16'd7777;
    end
    else data_store_reg7 <= data_value7;
  end
end
assign bus_x88_i = arg2 ^ arg3;
assign bus_x89_i = arg4 | arg6 nor arg5;
```
Our Solution

- Automatically identify malicious circuits in third-party hardware design IP
  - Engineers read few lines instead of thousands or millions

```verilog
class assign bus_x87_i = arg0 & arg1;
always @(posedge clk) begin
  if (rst) data_store_reg7 <= 16'b0;
  else begin
    if (argcarry_i37 == 16'hbacd0013) begin
      data_store_reg7 <= 16'd7777;
    end
    else data_store_reg7 <= data_value7;
  end
end
assign bus_x88_i = arg2 ^ arg3;
assign bus_x89_i = arg4 | arg6 nor arg5;
```
Currently Undergoing Testing
Overview

• Motivation
  • Hardware can be evil, don’t live in denial

• Key Observation
  • Evil hardware is stealthy

• Algorithm
  • Rank gates by degree of stealth

• Results
  • No false negatives, pragmatic and effective

• The Future of FANCI
  • How would we attack our own tool?

• Conclusions
  • Can we really use this tool today? (Spoiler: Yes)
Backdoors: Fact #1

Backdoor = Trigger + Payload

AES Key Stealing  Ciphertext  Key Exfiltration

0xba5eba11  d(0xba5eba11)
Backdoors: Fact #1

Backdoor = Trigger + Payload

AES Key Stealing  Ciphertext  Key Exfiltration

0xba5eba11
Backdoors: Fact #2

Stealth = Power
Backdoors: Fact #3

Validation != Security
What FANCI Does

- We need to catch stealthy circuits that validation is not able to catch
What FANCI Does

All Code

Stealthy Code

Malicious Code
Identifying Stealthy Code

- We propose a new quantitative measure of stealth
  - We rank wires in a circuit by stealth value
- Any wire is connected to many other wires
  - *Stealth* value is computed from the *control* values of all the wires it's connected to

**Example Histograms of Stealth Values**

![Graphs showing stealth value distribution](image-url)
Identifying Stealthy Code

• We propose a new quantitative measure of stealth
  • We rank wires in a circuit by stealth value

• Any wire is connected to many other wires
  • **Stealth** value is computed from the **control** values of all the wires its connected to

---

**Example Histograms of Stealth Values**
Defining Control

How often does an input matter?

Out = f(A, B, C)
How often does an input matter?

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>OUT</th>
<th>C Matters?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>YES</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>YES</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>NO</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>NO</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>NO</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>NO</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>YES</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>YES</td>
</tr>
</tbody>
</table>
How often does an input matter?

Control = #Observed / Total = 2/4 = 0.5

The effect of C on OUT is 0.5
Larger Circuits

Control = #Observed / Total = 2/16 = 0.125

A B C D E OUT E Matters?

1 1 1 1 1 1 1 0 1 1
1 1 1 1 1 1 0 0 1 1
1 1 1 1 1 0 0 1 1
1 1 1 1 0 0 1 1

YES

NO

32 Rows
16 Pairs

YES

YES
Example: 4-to-1 Mux

- Consider a real circuit (4-to-1 multiplexer)
  - How can we measure control?
Example: 4-to-1 Mux

- When is M dependent on A?
  - When $S_1 = S_2 = 0$ (one fourth of cases)
  - Total effect = 0.25
Example: 4-to-1 Mux

- M is dependent on $S_1$ and sometimes affected
  - When $A$ is different from $C$ (and $S_2 = 0$)
  - When $B$ is different from $D$ (and $S_2 = 1$)
  - One half of cases (total effect = 0.5)
Does This Look Suspicious?

<table>
<thead>
<tr>
<th>M</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>S₁</th>
<th>S₂</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
<td>0.50</td>
<td>0.50</td>
</tr>
</tbody>
</table>

A 0.25
B 0.25
C 0.25
D 0.25
S₁ 0.5
S₂ 0.5
Does This Look Suspicious?

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>S₁</th>
<th>S₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
<td>0.50</td>
<td>0.50</td>
</tr>
</tbody>
</table>

Definitely not
Does This Look Suspicious?

Definitely yes
Does This Look Suspicious?

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>$S_1$</th>
<th>$S_2$</th>
<th>$S_{3-66}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
<td>2$^{-65}$</td>
<td>0.50</td>
<td>0.50</td>
<td>2$^{-63}$</td>
</tr>
</tbody>
</table>

 Definitely yes

Just checking the min value is often not enough.

Better heuristics are needed to evaluate the vector.

Logic
Computing Stealth From Control

Mean (M) = (2.0 / 6) = 0.33
Median (M) = 0.25
Triviality (M) = 0.50
Mean(M) = (2.0 / 71) = 0.03
Median(M) = 2^-63
Triviality(M) = 0.50
Computing Stealth From Control

Mean(M) = \(2.0 / 71\) = 0.03
Median(M) = \(2^{-63}\)
Triviality(M) = 0.50

Triviality detects more triggers.
Mean/median detect more payloads.
### Optimization: Sampling

#### 2^N Rows

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>OUT</th>
<th>E Matters?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>YES</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>NO</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>YES</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>YES</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>YES</td>
</tr>
</tbody>
</table>
Optimization: Sampling

Approx. Effect = #Observed / #Sampled = $\frac{1}{2} = 0.5$

2^N Rows

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>OUT</th>
<th>E Matters?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>YES</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>NO</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>YES</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>YES</td>
</tr>
</tbody>
</table>
Results

• Stealth metrics are effective for existing benchmarks
  • No false negatives for TrustHub benchmarks

• Effective even on large designs
  • Able to process full (academic) microprocessor cores

• Efficient enough for modern designs
  • About 1 day to process an average sized module

• Can catch well-hidden backdoors
  • 100% coverage against “stealthy, malicious backdoors” (SSP 2011)
Effectiveness On TrustHub

False Positive Rates for TrustHub Benchmarks

- Mean
- Median
- Both
- Triviality

False Positives as a Percentage of Total Wires

TrusHub Benchmark Group:
- RS232
- s15850
- s35932
- s38417
How Would We Attack FANCI?

- Frequent-Action Backdoor
  - No stealth, requires incompetent/non-existent validation engineers

- False Positive Flooding
  - Contrived design, requires naïve integration engineer

- Pathological Pipeline (State Explosion) Backdoor
  - Contrived design, requires naïve integration engineer

- Foundry (Physical/Parametric) Backdoor
  - Malicious device from benign design, requires malicious foundry
Security Assurances

- Zero false negatives *so far*
  - Mathematical connection exists between stealth and validation

- FANCI flags wires if and only if they are stealthy
  - Static and not probabilistic or dynamic

- Can operate on digital, synchronous design IP
  - Source code or gatelists

- Can achieve design-side security with minimal validation
  - Works well with current state of practice
The Big Picture: Hardware Security

RTL Code → Design Synthesis → Physical Synthesis → Device Fabrication → Device Audit
The Big Picture: Hardware Security

- **Design Attacks**
  - **Insiders**
    - Hicks et al., 2010, Waksman et al., 2010 and 2011
  - **Third-Party IP**
    - This Talk

- **CAD Tool Attacks**
  - **Automated Malicious Design IP**
    - This Talk

- **Foundry Attacks**
  - **Counterfeiting**
    - Chakraborty et al., 2008, Rajendran et al., 2012
  - **Malicious Injections**
    - Agrawal et al., 2007, Banga et al., 2008, Salmani et al., 2009, Next talk
Conclusions

• Hardware backdoors: A serious, immediate threat
  • Currently no way to certify trustworthiness
  • Causes tech. localization (increased costs)

• FANCI: Static analysis to identify suspicious circuits
  • Zero false negatives so far
  • Minimal reliance on validation personnel

• Current Status
  • Practical, ready for modern designs (e.g., AFRL, CSAW)
  • First hardware certification tool for trustworthy IP