Paper Review of ‘A2: Analog Malicious Hardware’

Bastian Schildknecht
Seminar in Computer Architecture 2019
ETH Zürich
A2: Analog Malicious Hardware

Kaiyuan Yang, Matthew Hicks, Qing Dong, Todd Austin, Dennis Sylvester

University of Michigan

Distinguished paper at IEEE Symposium on Security and Privacy 2016
Background
Why Secure Hardware Matters

A system with insecure hardware means an insecure system.
Why Secure Hardware Matters

A system with **insecure hardware** means an insecure system

How does a hardware attack work?
Typical Trigger Based Hardware Attacks
Typical Trigger Based Hardware Attacks

Rare, But Attacker Controllable Event
Typical Trigger Based Hardware Attacks

Victim Wire

RBACE signal

Trigger

Payload

Rare, But Attacker Controllable Event
Typical Trigger Based Hardware Attacks

Victim Wire

RBACE signal

Trigger

Payload

Rare, But Attacker Controllable Event
When to Implement a Hardware Attack

Digital Design → Netlist

Background
When to Implement a Hardware Attack

Digital Design → Netlist → Back-end Design → GDSII*  

*Graphic Database System II file
When to Implement a Hardware Attack

**Background**

- Digital Design
- Back-end Design
- Fabrication
- Netlist
- GDSII*
- Chip

*Graphic Database System II file*
When to Implement a Hardware Attack

Digital Design → Back-end Design → Fabrication → Testing & Verification

Netlist → GDSII* → Chip

*Graphic Database System II file
When to Implement a Hardware Attack

Expected to cost $20,000,000,000 by 2020 for the smallest technology node

*Graphic Database System II file
When to Implement a Hardware Attack

Expected to cost $20,000,000,000 by 2020 for the smallest technology node

*Graphic Database System II file
Where to Put a Hardware Attack
Where to Put a Hardware Attack

20-30% of chip area is unused

Example GDSII layout with free space
Where to Put a Hardware Attack

20-30% of chip area is unused

Example GDSII layout with free space
Where to Put a Hardware Attack

20-30% of chip area is unused

Mostly caused by routing constraints

Example GDSII layout with free space
Where to Put a Hardware Attack

20-30% of chip area is unused

 Mostly caused by routing constraints

Opens up possibility for attackers to embed malicious hardware

Example GDSII layout with free space
Problem
Issues With Existing Hardware Attacks
Issues With Existing Hardware Attacks

Digital Domain Hardware Attacks
Issues With Existing Hardware Attacks

Digital Domain Hardware Attacks
Rely on triggers based on tens to hundreds of logic gates
Issues With Existing Hardware Attacks

Digital Domain Hardware Attacks
Rely on triggers based on tens to hundreds of logic gates
Not very small and not stealthy
Issues With Existing Hardware Attacks

Digital Domain Hardware Attacks
Rely on triggers based on tens to hundreds of logic gates
Not very small and not stealthy

Process Reliability Trojans
Issues With Existing Hardware Attacks

Digital Domain Hardware Attacks
Rely on triggers based on tens to hundreds of logic gates
Not very small and not stealthy

Process Reliability Trojans
Modify the fabrication process to cause the entire chip to fail early
Issues With Existing Hardware Attacks

Digital Domain Hardware Attacks
Rely on triggers based on tens to hundreds of logic gates
Not very small and not stealthy

Process Reliability Trojans
Modify the fabrication process to cause the entire chip to fail early
Not controllable
Issues With Existing Hardware Attacks

<table>
<thead>
<tr>
<th>Problem</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital Domain Hardware Attacks</td>
<td>Rely on triggers based on tens to hundreds of logic gates</td>
</tr>
<tr>
<td></td>
<td>Not very small and not stealthy</td>
</tr>
<tr>
<td>Process Reliability Trojans</td>
<td>Modify the fabrication process to cause the entire chip to fail early</td>
</tr>
<tr>
<td></td>
<td>Not controllable</td>
</tr>
<tr>
<td>Dopant-Level Trojans</td>
<td></td>
</tr>
</tbody>
</table>

Digital Domain Hardware Attacks
Rely on triggers based on tens to hundreds of logic gates
Not very small and not stealthy

Process Reliability Trojans
Modify the fabrication process to cause the entire chip to fail early
Not controllable

Dopant-Level Trojans

Issues With Existing Hardware Attacks

Digital Domain Hardware Attacks
Rely on triggers based on tens to hundreds of logic gates
Not very small and not stealthy

Process Reliability Trojans
Modify the fabrication process to cause the entire chip to fail early
Not controllable

Dopant-Level Trojans
Change behaviour of existing circuits by tying logic gates to logic 0 or 1
Issues With Existing Hardware Attacks

Digital Domain Hardware Attacks
Rely on triggers based on tens to hundreds of logic gates
Not very small and not stealthy

Process Reliability Trojans
Modify the fabrication process to cause the entire chip to fail early
Not controllable

Dopant-Level Trojans
Change behaviour of existing circuits by tying logic gates to logic 0 or 1
Not controllable and not stealthy
Issues With Existing Hardware Attacks

**Digital Domain Hardware Attacks**
Rely on triggers based on tens to hundreds of logic gates
Not very small and not stealthy

**Process Reliability Trojans**
Modify the fabrication process to cause the entire chip to fail early
Not controllable

**Dopant-Level Trojans**
Change behaviour of existing circuits by tying logic gates to logic 0 or 1
Not controllable and not stealthy

**Parametric Trojans for Fault Injection**
Issues With Existing Hardware Attacks

Digital Domain Hardware Attacks
Rely on triggers based on tens to hundreds of logic gates
Not very small and not stealthy

Process Reliability Trojans
Modify the fabrication process to cause the entire chip to fail early
Not controllable

Dopant-Level Trojans
Change behaviour of existing circuits by tying logic gates to logic 0 or 1
Not controllable and not stealthy

Parametric Trojans for Fault Injection
Same as dopant-level trojans but rely on voltage fluctuations as a trigger
Issues With Existing Hardware Attacks

**Digital Domain Hardware Attacks**
Rely on triggers based on tens to hundreds of logic gates
Not very small and not stealthy

**Process Reliability Trojans**
Modify the fabrication process to cause the entire chip to fail early
Not controllable

**Dopant-Level Trojans**
Change behaviour of existing circuits by tying logic gates to logic 0 or 1
Not controllable and not stealthy

**Parametric Trojans for Fault Injection**
Same as dopant-level trojans but rely on voltage fluctuations as a trigger
Not remotely controllable
Can We Do Better?

Is there a better hardware attack that does not suffer from these issues?
Can We Do Better?

Is there a better hardware attack that does not suffer from these issues?

How can it work?
Goal
Goal of the Paper

Goal: Design a hardware attack that is
Goal of the Paper

Goal: Design a hardware attack that is

Very small
Goal of the Paper

Goal: Design a hardware attack that is

Very small

Controllable
Goal of the Paper

Goal: Design a hardware attack that is

Very small

Controllable

Stealthy
Threat Model for the Attack
Threat Model for the Attack

- Attack implemented at time of fabrication
Threat Model for the Attack

- Attack implemented at time of \textit{fabrication}
- The attacker has only access to a \textit{correctly implemented GDSII} file
Threat Model for the Attack

- Attack implemented at time of **fabrication**
- The attacker has only access to a **correctly implemented GDSII** file
- The attacker **cannot change dimensions** or move stuff around
Threat Model for the Attack

- Attack implemented at time of **fabrication**
- The attacker has only access to a **correctly implemented GDSII** file
- The attacker **cannot change dimensions** or move stuff around
- The attacker has **no knowledge over tests** conducted on the chip
Previous Approaches of Hardware Attacks

Digital Domain Hardware Attacks
Rely on triggers based on tens to hundreds of logic gates
Not very small and not stealthy

Process Reliability Trojans
Modify the fabrication process to cause the entire chip to fail early
Not controllable

Dopant-Level Trojans
Change behaviour of existing circuits by tying logic gates to logic 0 or 1
Not controllable and not stealthy

Parametric Trojans for Fault Injection
Same as dopant-level trojans but rely on voltage fluctuations as a trigger
Not remotely controllable
Enter The A2 Attack

Novelty
The A2 attack uses analog behaviour to mitigate these issues!
The A2 attack uses analog behaviour to mitigate these issues!

```plaintext
on_every(RBACE) do
    if(count == 12345) then
        do_attack()
    else
        count = count + 1
    done
```
Enter The A2 Attack

The A2 attack uses analog behaviour to mitigate these issues!

```plaintext
on_every(RBACE) do
  if(count == 12345) then
    do_attack()
  else
    count = count + 1
  done
```

RBACE = victim wire
The A2 attack uses analog behaviour to mitigate these issues!

```
on_every(RBACE) do
    if(count == 12345) then
        do_attack()
    else
        count = count + 1
    done
```
The A2 attack uses analog behaviour to mitigate these issues!

```python
on_every(RBACE) do
    if(count == 12345) then
        do_attack()
    else
        count = count + 1
    end
end
```

How does this work?
Key Approach & Ideas
How does A2 work?

Victim Wire

Payload

Threshold Detector

Threshold Detector

Payload

current value

current charge
How does A2 work?

Key Approach & Ideas

Threshold Detector

Payload

Victim Wire

current value

current charge

Victim Wire
How does A2 work?

Key Approach & Ideas

Victim Wire

Payload

Threshold Detector

current value

current charge
How does A2 work?

Key Approach & Ideas

Victim Wire

Payload

Threshold Detector

current value

current charge
Mechanism in Detail
The Analog Trigger Circuit

Mechanism in Detail
The Analog Trigger Circuit
The Analog Trigger Circuit
The Analog Trigger Circuit
Design Challenge: Single Capacitor
Design Challenge: Single Capacitor

Small capacitors charge up to quickly
Small capacitors charge up to quickly
Design Challenge: Single Capacitor

Small capacitors charge up to quickly

- This results in the attack being too easy to trigger
Design Challenge: Single Capacitor

Small capacitors charge up to quickly
- This results in the attack being too easy to trigger

Large capacitors induce current spikes
Design Challenge: Single Capacitor

Small capacitors charge up to quickly
➔ This results in the attack being too easy to trigger

Large capacitors induce current spikes
Design Challenge: Single Capacitor

Small capacitors charge up to quickly
- This results in the attack being too easy to trigger

Large capacitors induce current spikes
- This makes it also easier to detect
Charge Sharing

Mechanism in Detail

VDD

Victim Wire

Victim Wire
Charge Sharing

- **VDD**
- **Victim Wire**
Charge Sharing

Mechanism in Detail

VDD

Victim Wire

Victim Wire
Charge Sharing

VDD

Victim Wire

Victim Wire
Charge Sharing

Victim Wire

Victim Wire

VDD
The Analog Trigger Circuit (Revised)
1. What can this trigger be used for?
The Analog Trigger Circuit (Revised)

1. What can this trigger be used for?
2. What do we connect it to?
1. Example Payload Circuit
1. Example Payload Circuit

Observation:
Observation:

Many processors de-escalate privilege stepwise after reset
1. Example Payload Circuit

Observation:
Many processors de-escalate privilege stepwise after reset

Idea:
1. Example Payload Circuit

Observation:

Many processors de-escalate privilege stepwise after reset

Idea:

Tap into reset wires of supervisor mode register
1. Example Payload Circuit
Privilege escalation by flipping the supervisor mode bit
1. Example Payload Circuit

Privilege escalation by flipping the supervisor mode bit

Active-low reset variant

Active-high reset variant
2. How to Find a Victim Wire
2. How to Find a Victim Wire

Observation:
2. How to Find a Victim Wire

**Observation:**

Need to find a software controllable wire with usually very low toggle rate
2. How to Find a Victim Wire

Observation:
Need to find a software controllable wire with usually very low toggle rate

Idea:
2. How to Find a Victim Wire

Observation:
Need to find a software controllable wire with usually very low toggle rate

Idea:
Simulate different programs to find wires with low toggle rates
2. How to Find a Victim Wire

Number of wires with a given toggle rate

- Proportion of Wires
- Toggle Rate

Graph showing the distribution of wires with a given toggle rate.
2. How to Find a Victim Wire

Choose wire with low toggle rate
2. How to Find a Victim Wire

Choose wire with low toggle rate e.g. divide by zero flag
2. How to Find a Victim Wire

Choose wire with low toggle rate
e.g. divide by zero flag

Choose threshold rate for this wire

Number of wires with a given toggle rate
2. How to Find a Victim Wire

Number of wires with a given toggle rate when the attack is running

![Graph showing the number of wires with a given toggle rate. The graph has a y-axis labeled 'Proportion of Wires' ranging from 0 to 0.03 and an x-axis labeled 'Toggle Rate' ranging from 0 to 0.25. There is a highlighted area labeled 'Threshold' indicating a specific range of toggle rates.]
2. How to Find a Victim Wire

Number of wires with a given toggle rate when the attack is running

Proportion of Wires

Threshold

Attacker toggles the wire frequently
2. How to Find a Victim Wire

Mechanism in Detail

Number of wires with a given toggle rate when the attack is running

Threshold

Attacker toggles the wire frequently
Attack gets triggered
Controlling the Attack From Software

**Attack Code Example**

```c
/* Victim wire is divide by zero flag */
while attack_success == 0 do
  i ← 0
  while i < 500 do
    z ← 1/0
    i ← i + 1
  end while
  if test_privileges() == 1 then
    attack_success ← 1
  end if
end while
```

Analog domain and digital domain of A2
Key Results
Methodology
Methodology

How the attack was evaluated:
Methodology

How the attack was evaluated:

1. Verification of design in simulation on 65nm CMOS in SPICE
Methodology

How the attack was evaluated:

1. Verification of design in simulation on 65nm CMOS in SPICE

2. Implementation and verification of design in a real processor
Methodology

How the attack was evaluated:

1. Verification of design in simulation on 65nm CMOS in SPICE

2. Implementation and verification of design in a real processor

3. Comparison of the results from 1. and 2.
Methodology

How the attack was evaluated:

1. Verification of design in simulation on 65nm CMOS in SPICE
2. Implementation and verification of design in a real processor
3. Comparison of the results from 1. and 2.
4. Assessing detectability
Implementation in a Real Chip

OpenRISC 1200 Processor

Main Memory
128KB SRAM

OR1200 Core

IO Drivers and Pads
Implementation in a Real Chip

OpenRISC 1200 Processor
Implementation in a Real Chip

OpenRISC 1200 Processor
Implementation in a Real Chip

OpenRISC 1200 Processor

Includes standalone trigger testing structure

Key Results
Implementation in a Real Chip

OpenRISC 1200 Processor

Includes stand-alone trigger testing structure

Main Memory 128KB SRAM

OR1200 Core

Uses only 0.08% of the total area!
Goal of the Paper

Goal: Design a hardware attack that is

Very small

Controllable

Stealthy
Goal of the Paper

Goal: Design a hardware attack that is

- Very small ✔
- Controllable
- Stealthy
Verification in the OR1200 Processor

Testing setup
Verification in the OR1200 Processor

Circuits tested under temperature, clock frequency and voltage variations
Verification in the OR1200 Processor

Circuits tested under temperature, clock frequency and voltage variations

Tested on multiple chips
Verification in the OR1200 Processor

Circuits tested under temperature, clock frequency and voltage variations

Tested on multiple chips

Trigger and retention times measured using the separate testing structure
Test Results of Real Chip Implementation
Test Results of Real Chip Implementation

Attacks in the chips are:
Test Results of Real Chip Implementation

Attacks in the chips are:

Robust against manufacturing variations
Test Results of Real Chip Implementation

Attacks in the chips are:

Robust against manufacturing variations

Robust against supply voltage fluctuations
Test Results of Real Chip Implementation

Attacks in the chips are:

Robust against manufacturing variations

Robust against supply voltage fluctuations

Robust against temperature changes
## Comparison to Simulation

<table>
<thead>
<tr>
<th>Trigger Circuit</th>
<th>Toggle Rate (MHz)</th>
<th>Measured (10 chip avg)</th>
<th>Simulated (Typical corner)</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o IO device</td>
<td>120.00</td>
<td>7.4</td>
<td>7</td>
</tr>
<tr>
<td>w/o IO device</td>
<td>34.29</td>
<td>8.4</td>
<td>8</td>
</tr>
<tr>
<td>w/o IO device</td>
<td>10.91</td>
<td>11.6</td>
<td>10</td>
</tr>
</tbody>
</table>
## Comparison to Simulation

<table>
<thead>
<tr>
<th>Trigger Circuit</th>
<th>Toggle Rate (MHz)</th>
<th>Measured (10 chip avg)</th>
<th>Simulated (Typical corner)</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o IO device</td>
<td>120.00</td>
<td>7.4</td>
<td>7</td>
</tr>
<tr>
<td>w/o IO device</td>
<td>34.29</td>
<td>8.4</td>
<td>8</td>
</tr>
<tr>
<td>w/o IO device</td>
<td>10.91</td>
<td>11.6</td>
<td>10</td>
</tr>
</tbody>
</table>
**Comparison to Simulation**

**Trigger times in cycles**

<table>
<thead>
<tr>
<th>Trigger Circuit</th>
<th>Toggle Rate (MHz)</th>
<th>Measured (10 chip avg)</th>
<th>Simulated (Typical corner)</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o IO device</td>
<td>120.00</td>
<td>7.4</td>
<td>7</td>
</tr>
<tr>
<td>w/o IO device</td>
<td>34.29</td>
<td>8.4</td>
<td>8</td>
</tr>
<tr>
<td>w/o IO device</td>
<td>10.91</td>
<td>11.6</td>
<td>10</td>
</tr>
</tbody>
</table>

Comparison shows that simulation has good enough accuracy to fabricate precise and controllable attacks!
Goal of the Paper

Goal: Design a hardware attack that is

Very small ✓

Controllable

Stealthy
Goal of the Paper

Goal: Design a hardware attack that is

Very small ✓
Controllable ✓
Stealthy
How to Detect a Hardware Attack
How to Detect a Hardware Attack

Side Channel Information
How to Detect a Hardware Attack

Side Channel Information
Temperature

Key Results
How to Detect a Hardware Attack

Side Channel Information
Temperature
Power requirements
How to Detect a Hardware Attack

Side Channel Information
Temperature
Power requirements
Electromagnetic measurements
How to Detect a Hardware Attack

Side Channel Information
Temperature
Power requirements
Electromagnetic measurements
Detects attacks that get hot or use much power
How to Detect a Hardware Attack

Side Channel Information
- Temperature
- Power requirements
- Electromagnetic measurements
  Detects attacks that get hot or use much power

Adding Sensors
How to Detect a Hardware Attack

Side Channel Information
- Temperature
- Power requirements
- Electromagnetic measurements
  Detects attacks that get hot or use much power

Adding Sensors
- Measure signal propagation delays

Key Results
How to Detect a Hardware Attack

Side Channel Information
Temperature
Power requirements
Electromagnetic measurements
Detects attacks that get hot or use much power

Adding Sensors
Measure signal propagation delays
Detects attacks that add logic to wires

Key Results
How to Detect a Hardware Attack

**Side Channel Information**
- Temperature
- Power requirements
- Electromagnetic measurements
  - Detects attacks that get hot or use much power

**Adding Sensors**
- Measure signal propagation delays
  - Detects attacks that add logic to wires

**Visual Inspection**
How to Detect a Hardware Attack

Side Channel Information
- Temperature
- Power requirements
- Electromagnetic measurements
  Detects attacks that get hot or use much power

Adding Sensors
- Measure signal propagation delays
  Detects attacks that add logic to wires

Visual Inspection
- Delayering the chip
How to Detect a Hardware Attack

Side Channel Information
Temperature
Power requirements
Electromagnetic measurements
Detects attacks that get hot or use much power

Adding Sensors
Measure signal propagation delays
Detects attacks that add logic to wires

Visual Inspection
Delaying the chip
Inspection via scanning electron microscope
How to Detect a Hardware Attack

Side Channel Information
Temperature
Power requirements
Electromagnetic measurements
Detects attacks that get hot or use much power

Adding Sensors
Measure signal propagation delays
Detects attacks that add logic to wires

Visual Inspection
Delaying the chip
Inspection via scanning electron microscope
Detects attacks that are big
How to Detect a Hardware Attack

Side Channel Information
Temperature
Power requirements
Electromagnetic measurements
Detects attacks that get hot or use much power

Adding Sensors
Measure signal propagation delays
Detects attacks that add logic to wires

Visual Inspection
Delaying the chip
Inspection via scanning electron microscope
Detects attacks that are big

Functional Testing
How to Detect a Hardware Attack

**Side Channel Information**
- Temperature
- Power requirements
- Electromagnetic measurements
  Detects attacks that get hot or use much power

**Visual Inspection**
- Delayering the chip
- Inspection via scanning electron microscope
  Detects attacks that are big

**Adding Sensors**
- Measure signal propagation delays
  Detects attacks that add logic to wires

**Functional Testing**
- Test for unexpected behaviour
How to Detect a Hardware Attack

**Side Channel Information**
- Temperature
- Power requirements
- Electromagnetic measurements
  
  Detects attacks that get hot or use much power

**Visual Inspection**
- Delayering the chip
- Inspection via scanning electron microscope
  
  Detects attacks that are big

**Adding Sensors**
- Measure signal propagation delays
  
  Detects attacks that add logic to wires

**Functional Testing**
- Test for unexpected behaviour
  
  Detects some attacks that change the circuit behaviour
How Stealthy is the Attack?

Key Results
How Stealthy is the Attack?

Can the attack be detected by side channels?
How Stealthy is the Attack?

Can the attack be detected by side channels?

Measuring of chip power consumption
How Stealthy is the Attack?

Can the attack be detected by side channels?
Measuring of chip power consumption
Simulating theoretical power usage of trigger circuit
How Stealthy is the Attack?

Can the attack be detected by *side channels*?

Measuring of chip power consumption

Simulating theoretical power usage of trigger circuit

Answer:
How Stealthy is the Attack?

Can the attack be detected by *side channels*?

Measuring of chip power consumption

Simulating theoretical power usage of trigger circuit

**Answer:**

The power requirements of the attack are well below normal fluctuations
Detection Mechanisms Evaded by A2

**Side Channel Information**
- Temperature
- Power requirements
- Electromagnetic measurements

**Adding Sensors**
- Measure signal propagation delays

**Visual Inspection**
- Delayering the chip
- Inspection via scanning electron microscope

**Functional Testing**
- Test for unexpected behaviour
Detection Mechanisms Evaded by A2

Side Channel Information
- Temperature
- Power requirements
- Electromagnetic measurements

Adding Sensors
- Measure signal propagation delays

Visual Inspection
- Delaying the chip
- Inspection via scanning electron microscope

Functional Testing
- Test for unexpected behaviour

Key Results
Maybe With On-Chip Sensors?
Maybe With On-Chip Sensors?

Can the attack be detected by **measuring propagation delays**?
Maybe With On-Chip Sensors?

Can the attack be detected by measuring propagation delays?

High accuracy simulation of trigger wire delays
Maybe With On-Chip Sensors?

Can the attack be detected by **measuring propagation delays**?

High accuracy simulation of trigger wire delays

Reset wires are typically asynchronous
Maybe With On-Chip Sensors?

Can the attack be detected by measuring propagation delays?

High accuracy simulation of trigger wire delays

Reset wires are typically asynchronous

Answer:
Maybe With On-Chip Sensors?

Can the attack be detected by measuring propagation delays?

High accuracy simulation of trigger wire delays

Reset wires are typically asynchronous

Answer:

For a $4\text{ns}$ clock period the delay change is only $0.33\%$ and well below process variation and noise
Detection Mechanisms Evaded by A2

Side Channel Information
- Temperature
- Power requirements
- Electromagnetic measurements

Adding Sensors
- Measure signal propagation delays

Visual Inspection
- Delayering the chip
- Inspection via scanning electron microscope

Functional Testing
- Test for unexpected behaviour

Key Results
Detection Mechanisms Evaded by A2

**Side Channel Information**
- Temperature
- Power requirements
- Electromagnetic measurements

**Adding Sensors**
- Measure signal propagation delays

**Visual Inspection**
- Delayering the chip
- Inspection via scanning electron microscope

**Functional Testing**
- Test for unexpected behaviour
And Visual Inspection?
And Visual Inspection?

Can the attack be found by looking at the chip?
And Visual Inspection?

Can the attack be found by looking at the chip?

A2 is as small as one gate and is almost identical to the other gates in a design.
And Visual Inspection?

Can the attack be found by looking at the chip?

A2 is as small as one gate and is almost identical to the other gates in a design. Difficult to distinguish one gate in a sea of hundreds of thousands of gates (or even more).
And Visual Inspection?

Can the attack be found by looking at the chip?

A2 is as small as one gate and is almost identical to the other gates in a design. Difficult to distinguish one gate in a sea of hundreds of thousands of gates (or even more).

Requires delayering to very low layers.
And Visual Inspection?

Can the attack be found by looking at the chip?

A2 is as small as one gate and is almost identical to the other gates in a design. Difficult to distinguish one gate in a sea of hundreds of thousands of gates (or even more).

Requires delayering to very low layers.

Answer:
And Visual Inspection?

Can the attack be found by looking at the chip?

A2 is as small as one gate and is almost identical to the other gates in a design.

Difficult to distinguish one gate in a sea of hundreds of thousands of gates (or even more).

Requires delayering to very low layers.

Answer:

A2 is unlikely to be found by visual inspection.
Detection Mechanisms Evaded by A2

**Side Channel Information**
- Temperature
- Power requirements
- Electromagnetic measurements

**Adding Sensors**
- Measure signal propagation delays

**Visual Inspection**
- Delayering the chip
- Inspection via scanning electron microscope

**Functional Testing**
- Test for unexpected behaviour

---

**Key Results**
Detection Mechanisms Evaded by A2

**Side Channel Information**
- Temperature
- Power requirements
- Electromagnetic measurements

**Adding Sensors**
Measure signal propagation delays

**Visual Inspection**
- Delayering the chip
- Inspection via scanning electron microscope

**Functional Testing**
Test for unexpected behaviour
What About Functional Testing?
What About Functional Testing?

Is the attack triggered during normal execution?
What About Functional Testing?

Is the attack triggered during normal execution?

Testing with five selected benchmark programs
What About Functional Testing?

Is the attack triggered during normal execution?

Testing with five selected benchmark programs

Testing over 6 different temperatures from -25°C to 100°C
What About Functional Testing?

Is the attack triggered during normal execution?

Testing with five selected benchmark programs

Testing over 6 different temperatures from -25°C to 100°C

Answer:
What About Functional Testing?

Is the attack triggered during normal execution?

Testing with five selected benchmark programs

Testing over 6 different temperatures from -25°C to 100°C

Answer:

The attack was not activated across all programs and temperatures
Detection Mechanisms Evaded by A2

**Side Channel Information**
- Temperature
- Power requirements
- Electromagnetic measurements

**Adding Sensors**
- Measure signal propagation delays

**Visual Inspection**
- Delaying the chip
- Inspection via scanning electron microscope

**Functional Testing**
- Test for unexpected behaviour

**Key Results**
Detection Mechanisms Evaded by A2

**Side Channel Information**
- Temperature
- Power requirements
- Electromagnetic measurements

**Adding Sensors**
- Measure signal propagation delays

**Visual Inspection**
- Delayering the chip
- Inspection via scanning electron microscope

**Functional Testing**
- Test for unexpected behaviour

**Key Results**
Detection Mechanisms Evaded by A2

Side Channel Information
- Temperature
- Power requirements
- Electromagnetic measurements

Adding Sensors
- Measure signal propagation delays

Visual Inspection
- Delayering the chip
- Inspection via scanning electron microscope

Functional Testing
- Test for unexpected behaviour

Key Results
Detection Mechanisms Evaded by A2

**Side Channel Information**
- Temperature
- Power requirements
- Electromagnetic measurements

**Adding Sensors**
- Measure signal propagation delays

**Visual Inspection**
- Delayering the chip
- Inspection via scanning electron microscope

**Functional Testing**
- Test for unexpected behaviour

**Key Results**

A2 is not easily detectable!
Goal of the Paper

Goal: Design a hardware attack that is

- Very small ✔
- Controllable ✔
- Stealthy
Goal of the Paper

Goal: Design a hardware attack that is

- Very small ✓
- Controllable ✓
- Stealthy ✓
Goal of the Paper

Goal: Design a hardware attack that is

- Very small ✓
- Controllable ✓
- Stealthy ✓
Possible Defenses Against A2
Possible Defenses Against A2

One possible defense against A2 could come in the form of *split manufacturing*:
Possible Defenses Against A2

One possible defense against A2 could come in the form of split manufacturing:

Subset of the chip design is fabricated in a trusted manufacturing facility
Possible Defenses Against A2

One possible defense against A2 could come in the form of **split manufacturing**:  
Subset of the chip design is fabricated in a trusted manufacturing facility

Very expensive
One possible defense against A2 could come in the form of **split manufacturing**:

Subset of the chip design is fabricated in a trusted manufacturing facility

**Very expensive**

**Difficult to do**, as wires can be reverse engineered and flip-flops are typically fabricated by the third party
Possible Defenses Against A2

One possible defense against A2 could come in the form of split manufacturing:

Subset of the chip design is fabricated in a trusted manufacturing facility

Very expensive

Difficult to do, as wires can be reverse engineered and flip-flops are typically fabricated by the third party

Needs a new type of defense!
Summary
Problem: Current hardware attacks have some inherent flaws, i.e., they are 1) big, 2) uncontrollable or 3) not stealthy enough
**Summary**

**Problem:** Current hardware attacks have some inherent flaws, i.e., they are 1) big, 2) uncontrollable or 3) not stealthy enough.

**Goal:** create a hardware attack that is small (i.e., requires as little as one gate) and stealthy (i.e., requires an unlikely trigger sequence before effecting a chip’s functionality) and controllable.
Summary

Problem: Current hardware attacks have some inherent flaws, i.e., they are 1) big, 2) uncontrollable or 3) not stealthy enough.

Goal: create a hardware attack that is small (i.e., requires as little as one gate) and stealthy (i.e., requires an unlikely trigger sequence before effecting a chip’s functionality) and controllable.

Key Idea:
- Construct a circuit that only uses 2 capacitors to siphon charge from nearby wires as they transition between digital values.
- When the capacitors are fully charged, deploy an attack that forces a victim flip-flop to the desired value.
Summary

Problem: Current hardware attacks have some inherent flaws, i.e., they are 1) big, 2) uncontrollable or 3) not stealthy enough.

Goal: create a hardware attack that is small (i.e., requires as little as one gate) and stealthy (i.e., requires an unlikely trigger sequence before effecting a chip’s functionality) and controllable.

Key Idea:
- Construct a circuit that only uses 2 capacitors to siphon charge from nearby wires as they transition between digital values.
- When the capacitors are fully charged, deploy an attack that forces a victim flip-flop to the desired value.

Key Results: 1) Implemented this attack in an OR1200 processor and fabricated a chip; 2) Experimental results show that the attack works efficiently; 3) The attack eludes activation by a diverse set of benchmarks; 4) the attack evades known defenses.
Strengths of the Paper
Strengths of the Paper

+ Shows a new type of hardware attack not seen before
Strengths of the Paper

+ Shows a new type of hardware attack not seen before
+ Real hardware implementation
Strengths of the Paper

+ Shows a new type of hardware attack not seen before
+ Real hardware implementation
+ Shows thorough testing of the attack
Strengths of the Paper

+ Shows a new type of hardware attack not seen before
+ Real hardware implementation
+ Shows thorough testing of the attack
+ Uses a strong and realistic threat model
Strengths of the Paper

+ Shows a new type of hardware attack not seen before
+ Real hardware implementation
+ Shows thorough testing of the attack
+ Uses a strong and realistic threat model
+ Assesses the possibility of an implementation in different architectures
Strengths of the Paper

+ Shows a new type of hardware attack not seen before
+ Real hardware implementation
+ Shows thorough testing of the attack
+ Uses a strong and realistic threat model
+ Assesses the possibility of an implementation in different architectures
+ Well written and relatively easy to understand
Strengths of the Paper

+ Shows a new type of hardware attack not seen before
+ Real hardware implementation
+ Shows thorough testing of the attack
+ Uses a strong and realistic threat model
+ Assesses the possibility of an implementation in different architectures
+ Well written and relatively easy to understand
+ Gives a history on previous work done in the field
Weaknesses & Limitations
Weaknesses & Limitations
Weaknesses & Limitations

- Does not give a concrete defense mechanism
Weaknesses & Limitations

- Does not give a concrete defense mechanism

- Cannot test hypothesis on other architectures due to cost and secrecy
Weaknesses & Limitations

- Does not give a concrete defense mechanism

- Cannot test hypothesis on other architectures due to cost and secrecy

- Contains a few typos
Thoughts & Ideas
Thought and Ideas

- Can this charge-pump mechanism be used for good purposes?
  - i.e. avoiding complicated state machines where precision is not as important
  - As was mentioned last week, maybe to prevent Rowhammer attacks?
Can this charge-pump mechanism be used for good purposes?
- i.e. avoiding complicated state machines where precision is not as important
- As was mentioned last week, maybe to prevent Rowhammer attacks?

Is this attack already used?
- I have not found any evidence that this attack is being used yet (please prove me wrong)
- I have found cases for other hardware trojans though, e.g. [1]
- Can you think of other cases of hardware attacks being used?

Thought and Ideas

- Can this charge-pump mechanism be used for good purposes?
  - i.e. avoiding complicated state machines where precision is not as important
  - As was mentioned last week, maybe to prevent Rowhammer attacks?

- Is this attack already used?
  - I have not found any evidence that this attack is being used yet (please prove me wrong)
  - I have found cases for other hardware trojans though, e.g. [1]
  - Can you think of other cases of hardware attacks being used?

- What has to be considered when applying this attack to other (smaller) technology nodes?

---

Some Interesting Follow-Up Papers
Some Interesting Follow-Up Papers

- Yumin Hou, Hu He, Kaveh Shamsi, Yier Jin, Dong Wu, Huaqiang Wu, "R2D2: Runtime reassurance and detection of A2 Trojan", Hardware Oriented Security and Trust (HOST) 2018 IEEE International
Some Interesting Follow-Up Papers

- Yumin Hou, Hu He, Kaveh Shamsi, Yier Jin, Dong Wu, Huaqiang Wu, "R2D2: Runtime reassurance and detection of A2 Trojan", Hardware Oriented Security and Trust (HOST) 2018 IEEE International

Some Interesting Follow-Up Papers

- Yumin Hou, Hu He, Kaveh Shamsi, Yier Jin, Dong Wu, Huaqiang Wu, "R2D2: Runtime reassurance and detection of A2 Trojan", Hardware Oriented Security and Trust (HOST) 2018 IEEE International Conference.


Some Interesting Follow-Up Papers

- Yumin Hou, Hu He, Kaveh Shamsi, Yier Jin, Dong Wu, Huaqiang Wu, "R2D2: Runtime reassurance and detection of A2 Trojan", Hardware Oriented Security and Trust (HOST) 2018 IEEE International Conference.


Open Discussion
Open Discussion

- How would you try to detect A2?
- How bad do you think is this type of attack?
- Can you think of a better attack?
- Do you think the shown follow-up papers solve the problem?
- Can the proposed mechanism be used for good?
- What are your thoughts on this paper?
- What do you think are the most important takeaways here?
Open Discussion

- How would you try to detect A2?
- How bad do you think is this type of attack?
- Can you think of a better attack?
- Do you think the shown follow-up papers solve the problem?
- Can the proposed mechanism be used for good?
- What are your thoughts on this paper?
- What do you think are the most important takeaways here?
Thank You For Your Attention!
Threshold Detector
Two possibilities for threshold detectors

Schmitt Trigger

Skewed Inverter
Threshold Detector

Two possibilities for threshold detectors

- Skewed inverter with fixed switching voltage

Schmitt Trigger

Skewed Inverter
Threshold Detector

Two possibilities for threshold detectors

- Skewed inverter with fixed switching voltage
- Schmitt trigger with hysteresis, i.e. high threshold on rising edge and low threshold on falling edge
Threshold Detector

Two possibilities for threshold detectors

- Skewed inverter with fixed switching voltage
- Schmitt trigger with hysteresis, i.e. high threshold on rising edge and low threshold on falling edge

Paper chooses Schmitt trigger as it extends trigger and retention time
Possibility: Chaining Triggers Together

- **Single stage trigger**
- Final Trigger = OA & OB
  - Either A or B triggers

- **Single stage trigger**
  - Final Trigger = OA | OB
  - Both A and B trigger

- **Single stage trigger**
  - Final Trigger = (OA & OB) | OC
  - One of A and B trigger, C trigger
Possibility: Chaining Triggers Together

- Triggers can be combined to form more complex trigger mechanisms
Possibility: Chaining Triggers Together

- Triggers can be combined to form more complex trigger mechanisms
- Can be used to construct well hidden multi-stage triggers
SPICE Simulation Results

- Trigger input
- Trigger output
- Cap Voltage

- Trigger Time: 240ns
- Retention Time: 0.8us
SPICE Using I/O Devices in 65nm CMOS
To mitigate gate leakage, I/O Device Cells can be used instead of normal standard cells.
To mitigate gate leakage, I/O Device Cells can be used instead of normal standard cells.

Results in more control over trigger and retention times.
To mitigate gate leakage, I/O Device Cells can be used instead of normal standard cells

Results in more control over trigger and retention times

Uses slightly more chip area
To mitigate gate leakage, I/O Device Cells can be used instead of normal standard cells.

- Results in more control over trigger and retention times
- Uses slightly more chip area
- Also simulated in 65nm low power CMOS
Stand-alone Testing Structure

CLK divider and duty cycle controller

Parameters From Scan Chain

Single stage trigger

COUNTER

To Scan Chain

To Scan Chain
Results Across 10 Chips (1V, 25°C)
Results Across 10 Chips (1V, 25°C)

- Shows number of chips which show a certain trigger time in cycles at different switching frequencies

(a) Distribution of analog trigger circuit using IO device

(b) Distribution of analog trigger circuit using only core device
Results Across 10 Chips (1V, 25°C)

- Shows number of chips which show a certain trigger time in cycles at different switching frequencies
- Also shows number of chips which show a certain retention time in µs
Results Across 10 Chips (1V, 25°C)

- Shows number of chips which show a certain trigger time in cycles at different switching frequencies
- Also shows number of chips which show a certain retention time in µs
- Shows robustness against manufacturing variations
Varying the Voltage
Varying the Voltage

- Shows the trigger time in cycles for a given voltage and frequency
Varying the Voltage

- Shows the trigger time in cycles for a given voltage and frequency
- Shows robustness across variations in the supply voltage
Varying the Temperature
Varying the Temperature

- Shows the trigger time in cycles for a given temperature and frequency
Varying the Temperature

- Shows the trigger time in cycles for a given temperature and frequency
- Shows robustness across variations in the ambient temperature
Varying the Temperature

- Shows the trigger time in cycles for a given temperature and frequency
- Shows robustness across variations in the ambient temperature
- The paper states that both single and two-stage attacks trigger in all 10 chips over 6 tested temperatures
Power consumption of the chip measured down to 1 μA at 1V and 25°C

<table>
<thead>
<tr>
<th>Program</th>
<th>Power (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standby</td>
<td>6.210</td>
</tr>
<tr>
<td>Basic math</td>
<td>23.703</td>
</tr>
<tr>
<td>Dijkstra</td>
<td>16.550</td>
</tr>
<tr>
<td>FFT</td>
<td>18.120</td>
</tr>
<tr>
<td>SHA</td>
<td>18.032</td>
</tr>
<tr>
<td>Search</td>
<td>21.960</td>
</tr>
<tr>
<td>Single-stage Attack</td>
<td>19.505</td>
</tr>
<tr>
<td>Two-stage Attack</td>
<td>22.575</td>
</tr>
<tr>
<td>Unsigned Division</td>
<td>23.206</td>
</tr>
</tbody>
</table>

Table III: Power consumption of our test Chip running a variety of benchmark programs.
Evaluation of Side Channel Information

- Power consumption of the chip measured down to 1 μA at 1V and 25°C
- Simulated power consumption of the trigger is 5.3 nW with I/O devices and 0.5 μW without I/O devices at maximum switching activity

<table>
<thead>
<tr>
<th>Program</th>
<th>Power (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standby</td>
<td>6.210</td>
</tr>
<tr>
<td>Basic math</td>
<td>23.703</td>
</tr>
<tr>
<td>Dijkstra</td>
<td>16.550</td>
</tr>
<tr>
<td>FFT</td>
<td>18.120</td>
</tr>
<tr>
<td>SHA</td>
<td>18.032</td>
</tr>
<tr>
<td>Search</td>
<td>21.960</td>
</tr>
<tr>
<td>Single-stage Attack</td>
<td>19.505</td>
</tr>
<tr>
<td>Two-stage Attack</td>
<td>22.575</td>
</tr>
<tr>
<td>Unsigned Division</td>
<td>23.206</td>
</tr>
</tbody>
</table>

Table III: Power consumption of our test Chip running a variety of benchmark programs.
Evaluation of Side Channel Information

- Power consumption of the chip measured down to 1 µA at 1V and 25°C
- Simulated power consumption of the trigger is 5.3 nW with I/O devices and 0.5 µW without I/O devices at maximum switching activity
- Well below normal power fluctuations

<table>
<thead>
<tr>
<th>Program</th>
<th>Power (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standby</td>
<td>6.210</td>
</tr>
<tr>
<td>Basic math</td>
<td>23.703</td>
</tr>
<tr>
<td>Dijkstra</td>
<td>16.550</td>
</tr>
<tr>
<td>FFT</td>
<td>18.120</td>
</tr>
<tr>
<td>SHA</td>
<td>18.032</td>
</tr>
<tr>
<td>Search</td>
<td>21.960</td>
</tr>
<tr>
<td>Single-stage Attack</td>
<td>19.505</td>
</tr>
<tr>
<td>Two-stage Attack</td>
<td>22.575</td>
</tr>
<tr>
<td>Unsigned Division</td>
<td>23.206</td>
</tr>
</tbody>
</table>

Table III: Power consumption of our test Chip running a variety of benchmark programs.
Evaluation of Side Channel Information

- Power consumption of the chip measured down to 1 μA at 1V and 25°C
- Simulated power consumption of the trigger is 5.3 nW with I/O devices and 0.5 μW without I/O devices at maximum switching activity
- Well below normal power fluctuations
- Temperature and propagation delays are nearly unaffected by A2 as it is as small as one gate

<table>
<thead>
<tr>
<th>Program</th>
<th>Power (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standby</td>
<td>6.210</td>
</tr>
<tr>
<td>Basic math</td>
<td>23.703</td>
</tr>
<tr>
<td>Dijkstra</td>
<td>16.550</td>
</tr>
<tr>
<td>FFT</td>
<td>18.120</td>
</tr>
<tr>
<td>SHA</td>
<td>18.032</td>
</tr>
<tr>
<td>Search</td>
<td>21.960</td>
</tr>
<tr>
<td>Single-stage Attack</td>
<td>19.505</td>
</tr>
<tr>
<td>Two-stage Attack</td>
<td>22.575</td>
</tr>
<tr>
<td>Unsigned Division</td>
<td>23.206</td>
</tr>
</tbody>
</table>

Table III: Power consumption of our test Chip running a variety of benchmark programs.
Is X86 Safe From A2?
Is X86 Safe From A2?

- The authors expect A2 to be easier to implement in X86 as in OR1200
Is X86 Safe From A2?

- The authors expect A2 to be easier to implement in X86 as in OR1200
- X86 has likely more possible target registers
Is X86 Safe From A2?

- The authors expect A2 to be easier to implement in X86 as in OR1200
- X86 has likely more possible target registers
- X86 has also likely more viable victim wires
Is X86 Safe From A2?

- The authors expect A2 to be easier to implement in X86 as in OR1200
- X86 has likely more possible target registers
- X86 has also likely more viable victim wires
- Due to the complexity of X86, A2 should also be more difficult to detect
Is X86 Safe From A2?

- The authors expect A2 to be easier to implement in X86 as in OR1200
- X86 has likely more possible target registers
- X86 has also likely more viable victim wires
- Due to the complexity of X86, A2 should also be more difficult to detect
- The only expected challenge is maintaining controllability over the many redundant functional units in X86