Hardware CWE™ Special Interest Group (SIG)

Chair: Bob Heinemann (MITRE)

Co-Chair: “Manna” Parbati Kumar Manna (Intel)

MITRE Team: Gananand Kini, Steve Christey Coley, Alec Summers

MITRE

September 12, 2025

Agenda

REMINDER: This meeting is being recorded.

1	CWE Gaps based on the Most Important Hardware Weaknesses data	Arun K and Hareesh K	40 min
2			

Housekeeping

- **Schedule:**
  - **Next Meeting October 10:**
    - **12:30 – 1:30 PM EST (16:30 – 17:30 UTC)**
    - **Microsoft Teams**
- **Contact: cwe@mitre.org**
- **Mailing List:** *hw-cwe-special-interest-group-sig-list@mitre.org*
- **Minutes from previous meetings available on our GitHub site:**
  - **https://github.com/CWE-CAPEC/hw-cwe-sig**

Announcements

- **Most Important Hardware Weaknesses List published (https://cwe.mitre.org/topHW/index.html)**
- **CWE 4.18 released!**
  - MIHW View introduced (View-1432)
  - Expert Insights Category introduced (Category-1433)
- **CWE Content Development Repository (CDR) is now fully public.**
  - Enables the broader community to view, track, and contribute to entries submissions.
  - Content suggestions begin with the CWE Submission Form.
  - CDR can be accessed here:
    - https://github.com/CWE-CAPEC/CWE-Content-Development-Repository/

General Call for Topics

- Any news items to share with the group?
- Any high priority topics we should address today?
- Future topics?

CWE Gaps identified from Most Important Hardware Weaknesses data

Arun K and Hareesh K

Most Important HW CWE Data: Gap issues

- **HW CWE mapping review: Gap issues**
  - CWE-Gap: Used for issues that were HW issues but did not have a good matching CWE in existing HW CWEs
  - 15 issues were marked as CWE Gap in the final analysis
  - Sharing details on these and related CVEs or papers, advisories

- **Goal**
  - Review within SIG,
    - Do these need new CWEs or how these should be mapped using existing CWEs?
  - Invite SIG members to submit write-ups for any issues needing new CWE entries

https://docs.google.com/spreadsheets/d/1pTpzpT9WeD0X96fsn5ojeKtB5JQa8hJ9/edit?gid=347615923#gid=347615923 , Combined list, (CWE Final == Gap)

HW Fabric Protocol Violations

CWE Gap (Root cause)	Weakness details	CVE/ Advisory reference
Fabric protocol violations leading to security issues	1. Silent transaction drops triggered by protocol violations & errors 2. Unintended Memory Alterations through violations: transaction ordering rules, concurrent transactions 3. Stale data leak on Read transactions due to violations or read response on errors.	eXpect: On the Security Implications of Violations in AXI Implementations
Improper handling of malformed packets on HW interface	CVE-2024-47667: When an inbound PCIe TLP spans more than two internal AXI 128-byte bursts, the bus may corrupt the packet payload. This issue affects inbound reads only. Outbound transactions are not affected. The corrupt data may cause associated applications or the processor to hang	https://www.ti.com/lit/er/sprz452i/sprz452i.pdf

- **Protocol Violations**
  - Silent transaction drops, Stale data leak, transaction ordering violations
    - These will need to have specific CWEs within Peripherals, On-chip Fabric, and Interface/IO Problems - *(1203)* Category
    - CWE-1429: Missing Security-Relevant Feedback for Unexecuted Operations in Hardware Interface : Maps for Silent transaction drops

Incorrect Address Translation or Validation

CWE Gap (Root cause)	Weakness details	CVE/ Advisory reference
Improper/ Insufficient address validation/ translation	CVE-2021-26337: Insufficient DRAM address validation in System Management Unit (SMU) may result in a DMA read from invalid DRAM address to SRAM CVE-2023-20533: Insufficient DRAM address validation in System Management Unit (SMU) may allow an attacker to read/write from/to an invalid DRAM address CVE-2023-20566: Improper address validation in ASP with SNP enabled may potentially allow an attacker to compromise guest memory integrity. CVE-2023-20584: IOMMU improperly handles certain special address ranges with invalid device table entries (DTEs)	amd-sb-4002 amd-sb-1027 amd-sb-5001 amd-sb-3003

- **Core to Memory or IO address translation and validation**
  - Can be in MPU, Memory Controller, IOMMU etc.

Improperly Implemented Core Instructions

CWE Gap (Root cause)	Weakness details	CVE/ Advisory reference
Improper flow control in core instruction processing	CVE-2021-35465: On affected CPUs, if the VLLDM instruction is abandoned due to an exception (for example an interrupt) when it is partially completed, it is possible for subsequent Non-secure handler (which may be a tail chained handler from the original exception) to access and modify the partial restored register values.	https://developer.arm.com/documentation/110365/1-0/?lang=en

- **Core ISA instructions can have implementation issues**
  - This specific one is related flow control for handling exception before instruction processing is complete
  - Existing CWE that might map: Sequence of Processor Instructions Leads to Unexpected Behavior - *(1281)*
  - Does it make sense to detail specific weakness scenarios in instruction processing flows?

Others

- **These are more side-channel and Physical attack vectors**
  - Previously discussed that new CWEs are not needed for new attack paths

- **Side-Channel exposures**
  - Infrared:
    - HomeSpy: The Invisible Sniffer of Infrared Remote Control of Smart TVs (USENIX 23)
  - Covert channel based on cache timing
    - Side-Channel Attacks on Intel Optane Persistent Memory (USENIX 23)
  - Acoustic:
    - Side Eye: Characterizing the Limits of POV Acoustic Eavesdropping from Smartphone Cameras with Rolling Shutters and Movable Lenses (Security and Privacy 2023)
    - mmEcho: A mmWave-based Acoustic Eavesdropping Method (Security and Privacy 2023)

- **Physical attacks**
  - Magnetic injection
    - MagBackdoor: Beware of Your Loudspeaker as A Backdoor For Magnetic Injection Attacks (Security and Privacy 2023)
  - Laser
    - PLA-LiDAR: Physical Laser Attacks against LiDAR-based 3D Object Detection in Autonomous Vehicle (Security and Privacy 2023)
    - The Invisible Polyjuice Potion: An Effective Physical Adversarial Attack Against Face Recognition (ACM CCS 2024)

Next Meeting (October 10)

CWE@MITRE.ORG

- **Mailing List:** *hw-cwe-special-interest-group-sig-list@mitre.org*
  - ***NOTE: All mailing list items are archived publicly at:***
    - *https://www.mail-archive.com/hw-cwe-special-interest-group-sig-list@mitre.org/*
- **What would members of this body like to see for the next HW SIG agenda?**
- **Questions, Requests to present? Please let us know.**

Backup

Working Exercise

- Last month we the group agreed that we should look at existing memory weaknesses and update to include HW concerns.
- Let's review CWE-125: Out-of-bounds Read to see what we would need to update to make it inclusive of hardware.
- https://docs.google.com/document/d/18USrLkoXb8iLAREutRPAG8I7iEpL61wt5g0QTdCn-Vs/edit?usp=sharing

Memory Access Related Weaknesses

Jason Oberg

Introduction

Cycuity

Overview

- Several CWEs exist related to out-of-bound (OOB) buffer access that are not in the hardware view.
- In hardware, buffers manifest themselves as memories, register arrays, etc.

Impact of OOB Buffer Access in HW

- OOB access to buffers in RTL simulation results in undefined behavior (usually an ‘X’ for reads or no changes for writes)
- In real synthesized logic, data *will* be read/written and the actual result of that may be unknown to the hardware designer.

Proposal

- Move relevant CWEs to the hardware view or provide a comparable CWE in HW view

Memory Related CWEs that are not in HW View

Cycuity

CWE ID	Name	Description	HW Consequences
CWE-125	**Out-of-bounds Read**	The product reads data past the end, or before the beginning, of the intended buffer.	• In simulation, OOB read returns ‘X’ and is undefined • In a synthesized circuit, they OOB read could expose sensitive information
CWE-124	**Buffer Underwrite ('Buffer Underflow')**	The product writes to a buffer using an index or pointer that references a memory location prior to the beginning of the buffer.	• In simulation, an OOB write is essentially a “no-op” and the buffer isn’t updated • In a synthesized circuit, the OOB write may compromise the integrity of the buffer
CWE-787	**Out-of-bounds Write**	The product writes data past the end, or before the beginning, of the intended buffer.	• In simulation, an OOB write is essentially a “no-op” and the buffer isn’t updated • In a synthesized circuit, the OOB write may compromise the integrity of the buffer
CWE-786	**Access of Memory Location Before Start of Buffer**	The product reads or writes to a buffer using an index or pointer that references a memory location prior to the beginning of the buffer.	• Similar impact as above

Hardware Example:
CWE-125 Out-of-bounds Read

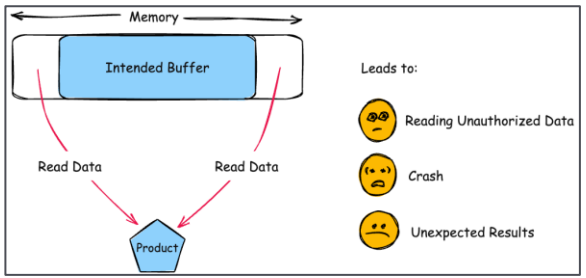
Cycuity

Description:

- The product reads data past the end, or before the beginning, of the intended buffer.

Hardware Consequences:

- OOB read in hardware is undefined and will return 'X' in simulation
- In a synthesized circuit, the data may be read from unexpected locations, potentially leaking data



https://cwe.mitre.org/data/definitions/125.html

```
1  module cwe787_cwe125 #(parameter WIDTH=4, parameter DEPTH=16) (
2      input [WIDTH-1 : 0] data,
3      input clk,
4      input we,
5      input [$clog2(DEPTH) - 1 : 0] waddr,
6      input [$clog2(DEPTH) - 1 : 0] raddr,
7      output reg [WIDTH-1 : 0] out
8
9
10     reg [WIDTH-1:0] mem [DEPTH-2 : 0];
11
12     always @(posedge clk) begin
13         if (we) mem[waddr] <= data;
14         out <= mem[raddr];
15     end
16
17 endmodule
```

`raddr` indexes between `h0 to `hF

Highest address of `mem` is `hE (one index short of `raddr`)

The read to `hF results in an OOB read

Hardware Example:
CWE-787 Out-of-bounds Write

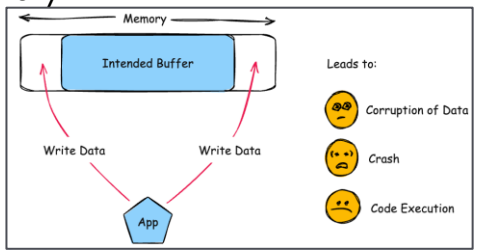
Cycuity

Description:

- The product writes data past the end, or before the beginning, of the intended buffer.

Hardware Consequences:

- OOB write in hardware is undefined and will basically not update the buffer/memory in *simulation*
- In real synthesized logic, a write may occur and could compromise the integrity of data in the memory.



https://cwe.mitre.org/data/definitions/787.html

```
1  module cwe787_cwe125 #(parameter WIDTH=4, parameter DEPTH=16) (
2      input [WIDTH-1 : 0] data,
3      input clk,
4      input we,
5      input [$clog2(DEPTH) - 1 : 0] waddr,
6      input [$clog2(DEPTH) - 1 : 0] raddr,
7      output reg [WIDTH-1 : 0] out
8  );
9
10     reg [WIDTH-1:0] mem [DEPTH-2 : 0];
11
12     always @(posedge clk) begin
13         if (we) mem[waddr] <= data;
14         out <= mem[raddr];
15     end
16
17 endmodule
```

`waddr` indexes between `h0 to `hF

Highest address of `mem` is `hE (one index short of `waddr`)

The write to `hF results in an OOB write

Questions for Discussion

Cycuity

- Have others in the community encountered similar issues in hardware?
- The contents of these CWEs are software centric. Would it make sense to update them to include some hardware examples before inclusion in the HW view?

Memory Weakness for HW Recap

Cycuity

- Out-of-bounds access weaknesses are well-defined in software, they are not as clearly defined in hardware.
- Differences in how out-of-bounds accesses behave between simulation and synthesized circuits may lead to security consequences.
- Out-of-bounds Reads may return an undefined value in simulation but in a synthesized circuit it could return data from an unexpected location.
- Out-of-bounds Writes in simulation are treated as no-ops, but in synthesized hardware they could modify unexpected locations.
- Existing memory weaknesses could be updated and added to the hardware view.
- It can be problematic to create HW version of existing entries and to do so would need an exceptional justification. – Steve C
- It will not be possible to distinguish CVEs referencing memory-related CWEs as HW or SW issues if existing memory CWEs are included in the HW CWE view. – Jason Oberg
- Hareesh had commented “We should also consider adding CWE-190: Integer Overflow or Wraparound”

Memory Related CWEs that are not in HW View

Cycuity

	CWE ID	Name	Description	HW Consequences
Jason O	CWE-125	**Out-of-bounds Read**	The product reads data past the end, or before the beginning, of the intended buffer.	• In simulation, OOB read returns 'X' and is undefined • In a synthesized circuit, they OOB read could expose sensitive information
	CWE-124	**Buffer Underwrite ('Buffer Underflow')**	The product writes to a buffer using an index or pointer that references a memory location prior to the beginning of the buffer.	• In simulation, an OOB write is essentially a "no-op" and the buffer isn't updated • In a synthesized circuit, the OOB write may compromise the integrity of the buffer
	CWE-787	**Out-of-bounds Write**	The product writes data past the end, or before the beginning, of the intended buffer.	• In simulation, an OOB write is essentially a "no-op" and the buffer isn't updated • In a synthesized circuit, the OOB write may compromise the integrity of the buffer
	CWE-786	**Access of Memory Location Before Start of Buffer**	The product reads or writes to a buffer using an index or pointer that references a memory location prior to the beginning of the buffer.	• Similar impact as above
Hareesh	CWE-190	**Integer Overflow or Wraparound**	The product performs a calculation that can produce an integer overflow or wraparound when the logic assumes that the resulting value will always be larger than the original value. This occurs when an integer value is incremented to a value that is too large to store in the associated representation. When this occurs, the value may become a very small or negative number.	