Hardware CWE™ Special Interest Group (SIG)

Gananand Kini, Bob Heinemann, Gage Hackford, Chris Lathrop, Steve Christey Coley, Alec Summers

MITRE

January 12, 2024

Agenda

REMINDER: This meeting is being recorded.

1	Community Items	Bob H	10 min
2	Microarchitectural Weaknesses Update	Bob H	10 Min
3	HW DEMOX's added so far with examples	Bob H	10 min
4	Composites and Chains Discussion	Bob H	20 Min

Housekeeping

- **Schedule:**
  - **Next Meeting: Feb 9**
    - **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

- **Reminder: December meeting was cancelled.**
- **CWE Content Development Repository (CDR) pilot now on GitHub! Currently invite only. Potential public release in early 2024.**
- **CWE 4.14 Release Planned for February 29.**
  - DEMOX's and Microarchitectural Weaknesses have been a priority for this release.

Open *Community Items*

HW CWE's With Missing:
DEMOX's, OBEX's and Mitigations

- **Missing Mitigations**
  - 4 HW CWEs are missing mitigations (No change)
- **Missing demonstrative examples (DEMOX)**
  - 15 HW CWEs missing demonstrative examples (down 1)
    - 1 added from Hack@DAC, CWE-440
    - Note: there are other DEMOXs from Hack@DAC but now adding DEMOXs to entries that have an existing DEMOX
- **Missing Observed Examples (OBEX)**
  - 61 CWEs do not have any observed examples (No Change)

https://github.com/CWE-CAPEC/hw-cwe-sig/issues

Discussion Items on GitHub

Resonant frequency weakness, proposal (Topic Lead: OPEN)	• https://github.com/CWE-CAPEC/hw-cwe-sig/issues/105
Covert Channel Coverage in HW View (Topic Lead: OPEN)	• https://github.com/CWE-CAPEC/hw-cwe-sig/issues/108
CWE Coverage of HW Cryptography (Topic Lead: OPEN)	• https://github.com/CWE-CAPEC/hw-cwe-sig/issues/7
Lifecycle-stage classification for HW CWEs –Dan DiMase (Topic Lead: OPEN)	• https://github.com/CWE-CAPEC/hw-cwe-sig/issues/4

Microarchitectural Weaknesses Update

Microarchitectural Weaknesses Update

4 new submissions being worked

- **CWE A: Exposure of Sensitive Information during Transient Execution**
  - CWE B: Exposure of Sensitive Information in Shared Microarchitectural Structures during Transient Execution
  - CWE C: Exposure of Sensitive Information caused by Incorrect Data Forwarding during Transient Execution
  - CWE D: Exposure of Sensitive Information caused by Shared Microarchitectural Predictor State that influences Transient Execution

Microarchitectural Weaknesses Update

- **We have received full submissions for CWE-A, CWE-B, CWE-C and CWE-D.**
- **Currently loaded into CDR and are at phase 8, Full Submission Received status.**
  - CWE-A_ES2309-a1334d32 (GH CDR Issue #35)
  - CWE-B_ES2309-f7b489ac (GH CDR Issue #34)
  - CWE-C_ES2309-78afd2b1 (GH CDR Issue #36)
  - CWE-D_ES2309-47154075 (GH CDR Issue #37)
- **Full submissions are out to the working group for comment.**
- **Comment period for working group is open to Friday, January 19th.**

Microarchitectural Weaknesses Update

- **Working group tentatively planned to meet on Monday, January 22nd.**
  - Collect final comments and hopefully get group approval to proceed with submissions
  - The group is pushing hard to get these part of release 4.14.

- **There was some initial concern about the potential for CWE-B to be mis-mapped to CWE-C in some cases. Proposal is currently out to address this through vulnerability mapping notes.**

- **Future item to discuss is path forward on CWE-1342: Information Exposure through Microarchitectural State after Transient Execution**

HW DEMOX’s Added with Examples

HACK@DAC Inspired DEMOX's

- **4.12 (3)**
  - CWE-1260, CWE-1262, CWE-1281
- **4.13 (10)**
  - CWE-1191, CWE-1220, CWE-1221, CWE-1231, CWE-1241
  - CWE-1243, CWE-1276, CWE-1300, CWE-1326, CWE-325
- **4.14 (9)**
  - CWE-1234, CWE-1239, CWE-1244 (Staged in DEV)
  - CWE-1329, CWE-1317, CWE-440 (Being added to DEV)
  - CWE-226, CWE-1298, CWE-1310 (In editorial review)

CWE-1234: Hardware Internal or Debug Modes Allow Override of Locks (DEMOX Introduction)

The following example code [REF-1375] is taken from the register lock security peripheral of the HACK@DAC'21 buggy OpenPiton SoC. It demonstrates how to lock read or write access to security-critical hardware registers (e.g., crypto keys, system integrity code, etc.). The configuration to lock all the sensitive registers in the SoC is managed through the reglk_mem registers. These reglk_mem registers are reset when the hardware powers up and configured during boot up. Malicious users, even with kernel-level software privilege, do not get access to the sensitive contents that are locked down. Hence, the security of the entire system can potentially be compromised if the register lock configurations are corrupted or if the register locks are disabled.

Example Language: **Verilog**

```
...
always @(posedge clk_i)
  begin
    if(~(rst_ni && ~jtag_unlock && ~rst_9))
      begin
        for (j=0; j < 6; j=j+1) begin
          reglk_mem[j] <= 'h0;
        end
      end
...
```

CWE-1234: Hardware Internal or Debug Modes Allow Override of Locks (Bad Code)

Example Language: **Verilog**

```
...
always @(posedge clk_i)
  begin
    if(~(rst_ni && ~jtag_unlock && ~rst_9))
      begin
        for (j=0; j < 6; j=j+1) begin
          reglk_mem[j] <= 'h0;
        end
      end
...
```

Note: This Bad Clock block repeated just for these slides. It only appears once in the CWE entry.

The example code [REF-1375] illustrates an instance of a vulnerable implementation of register locks in the SoC. In this flawed implementation [REF-1375], the reglk_mem registers are also being reset when the system enters debug mode (indicated by the jtag_unlock signal). Consequently, users can simply put the processor in debug mode to access sensitive contents that are supposed to be protected by the register lock feature.

CWE-1234: Hardware Internal or Debug Modes Allow Override of Locks (Good Code)

This can be mitigated by excluding debug mode signals from the reset logic of security-critical register locks as demonstrated in the following code snippet [REF-1376].

Example Language: **Verilog**

```
...
always @(posedge clk_i)
  begin
    if(~(rst_ni && ~rst_9))
      begin
        for (j=0; j < 6; j=j+1) begin
          reglk_mem[j] <= 'h0;
        end
      end
...
```

CWE Nit Bits

Composites and Chains
followed by discussion

Review: What is a weakness?

Weakness

- ***A condition*** in a software, firmware, hardware, or service component that, under certain circumstances, ***could contribute*** to the introduction of ***vulnerabilities***.
- Applies to the dimensions of behavior, property, resource, technology, or language.

Vulnerability

- **A flaw** in a software, firmware, hardware, or service component **resulting from a weakness** that can be exploited, causing a negative impact to the confidentiality, integrity, or availability of an impacted component or components.

A Weakness is a Root Cause for a Vulnerability

Interactions and Co-Occurrences of Weaknesses

- One may incorrectly infer that there is a one-to-one relationship between vulnerabilities and weaknesses.
- Assuming a one-to-one mapping can potentially lead to inconsistent mappings due to focusing on different parts of the same vulnerability[2].
- CWE's research has shown that vulnerabilities often can be described in terms of the interaction or co-occurrence of two or more weaknesses[1].
- The concepts of Chains and Composites were developed to describe these cases.

1. https://cwe.mitre.org/documents/glossary/index.html#Compound%20Element
2. https://cwe.mitre.org/documents/vulnerability_theory/intro.html#chap9

Chains[1]

- **Chain**
  - Sequence of two or more separate weaknesses that can be closely linked together.
  - One weakness, X, can directly create the conditions that are necessary to cause another weakness, Y, to enter a vulnerable condition.
    - X is primary to Y and Y is resultant from X.
  - Chains can involve more than two weaknesses, and in some cases, they might have a tree-like structure.

- **Named Chains**
  - Common chains that are assigned their own CWE identifiers.

1. *https://cwe.mitre.org/data/reports/chains_and_composites.html#chains*

Chain Examples (Primary CWE)

CWE-1260: Improper Handling of Overlap Between Protected Memory Ranges

Weakness ID: 1260
Abstraction: Base
Structure: Simple

View customized information: Conceptual | Operational | Mapping Friendly | Complete | **Custom**

Description

The product allows address regions to overlap, which can result in the bypassing of intended memory protection.

Extended Description

Isolated memory regions and access control (read/write) policies are used by hardware to protect privileged software. Software cc
flexible and dynamically changeable memory management by system software.

If a software component running at lower privilege can program a memory address region to overlap with other memory regions u
The memory protection unit (MPU) logic can incorrectly handle such an address overlap and allow the lower-privilege software to r
overlap weakness can also be used to launch a denial of service attack on the higher-privilege software memory regions.

Relationships

Relevant to the view "Research Concepts" (CWE-1000)

Nature	Type	ID	Name
ChildOf	P	284	Improper Access Control
CanPrecede		119	Improper Restriction of Operations within the Bounds of a Memory Buffer

Relevant to the view "Hardware Design" (CWE-1194)

Nature	Type	ID	Name
MemberOf	C	1198	Privilege Separation and Access Control Issues

Chain Examples (Resultant CWE)

CWE-119: Improper Restriction of Operations within the Bounds of a Memory Buffer

Weakness ID: 119
Abstraction: Class
Structure: Simple

View customized information: Conceptual | Operational | Mapping Friendly | Complete | Custom

For users who want to customize what details are displayed.

Description

The product performs operations on a memory buffer, but it can read from or write to a memory location that is outside of the intended bound

Extended Description

Certain languages allow direct addressing of memory locations and do not automatically ensure that these locations are valid for the memory l performed on memory locations that may be associated with other variables, data structures, or internal program data.

As a result, an attacker may be able to execute arbitrary code, alter the intended control flow, read sensitive information, or cause the system

Relationships

Relevant to the view "Research Concepts" (CWE-1000)

Nature	Type	ID	Name
ChildOf	C	118	Incorrect Access of Indexable Resource ('Range Error')
ParentOf	B	120	Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')
• • •			
CanFollow	B	128	Wrap-around Error
CanFollow	V	129	Improper Validation of Array Index
CanFollow	B	131	Incorrect Calculation of Buffer Size
CanFollow	B	190	Integer Overflow or Wraparound
CanFollow	B	193	Off-by-one Error
CanFollow	V	195	Signed to Unsigned Conversion Error
CanFollow	B	839	Numeric Range Comparison Without Minimum Check
CanFollow	B	843	Access of Resource Using Incompatible Type ('Type Confusion')
CanFollow	B	1257	Improper Access Control Applied to Mirrored or Aliased Memory Regions
CanFollow	B	1260	Improper Handling of Overlap Between Protected Memory Ranges
CanFollow	B	1339	Insufficient Precision or Accuracy of a Real Number

Named Chain Example

CWE-680: Integer Overflow to Buffer Overflow

Weakness ID: 680
Abstraction: Compound
Structure: Chain

View customized information: Conceptual | Operational | Mapping Friendly | **Complete** | Custom

Description

The product performs a calculation to determine how much memory to allocate, but an integer ove

Chain Components

Nature	Type	ID	Name
StartsWith	B	190	Integer Overflow or Wraparound
FollowedBy	C	119	Improper Restriction of Operations within the Bounds of a Memory Bu

Relationships

Relevant to the view "Research Concepts" (CWE-1000)

Nature	*Type*	*ID*	*Name*
ChildOf	B	190	Integer Overflow or Wraparound

Composites[1]

- **Composite**
  - Combination of two or more separate weaknesses that can create a vulnerability, but only if they all occur all the same time.
  - One weakness, X, can be "broken down" into component weaknesses Y and Z.
  - By eliminating any single component, a designer/implementer can prevent the composite from becoming exploitable.

1. *https://cwe.mitre.org/data/reports/chains_and_composites.html#chains*

Composite Example

CWE-689: Permission Race Condition During Resource Copy

Weakness ID: 689
Abstraction: Compound
Structure: Composite

View customized information: Conceptual | Operational | Mapping Friendly | Complete | Custom

Description

The product, while copying or cloning a resource, does not set the resource's permissions or access control until the copy is comp

Composite Components

Nature	Type	ID	Name
Requires	C	362	Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')
Requires	C	732	Incorrect Permission Assignment for Critical Resource

Relationships

Relevant to the view "Research Concepts" (CWE-1000)

Nature	Type	ID	Name
ChildOf	C	362	Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')

Discussion / Comments

- **What are your views on the applicability of Chains and Composites to hardware weaknesses?**
- **Do you see value in using these concepts to express hardware weaknesses?**
- **If not, are other concepts needed to better suit the hardware CWE space?**
- **What about their potential in addressing weaknesses that cross hardware-software boundaries?**

Next Meeting (Feb 9)

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

Covert Channel Coverage in CWE

COVID-Bit Research Item[1][2]

- **In 2022, researchers at Ben Gurion University in Israel developed a new data exfiltration method for air-gapped systems called COVID-bit.**
- **Malware generates electromagnetic radiation in the 0-60 kHz frequency band (assumes Malware got there somehow).**
- **EM emissions are generated by manipulating the workload of the CPU. Claims of indirect control SMPS.**
- **The electromagnetic radiation generated by this intentional process can be received from a distance using appropriate antennas.**

1. *https://thehackernews.com/2022/12/covid-bit-new-covert-channel-to.html?m=1*
2. *https://arxiv.org/abs/2212.03520*

Covert Channels and Side Channels

- Initial thought was that COVID-Bit could be a DEMOX for CWE-1300: Improper Protection of Physical Side Channels
- As HW SIG Members had correctly pointed out, COVID-Bit is about Covert Channels and NOT Side Channels
- Covert Channel (CC) / Side Channel (SC)
  - Intentional transmission (CC). Accidental transmission (SC) – *Ross Anderson* [1]
  - Adversary controls input and output (CC). Adversary can only read output (SC) – *Intel* [2]
  - Not an intended resource but exists due the application's behaviors. –*CWE-514 Notes* [3]
- If not CWE-1300 (SC), where would something like this map to in HW view?
- Closest we have is CWE-514: Covert Channels

1. https://www.cl.cam.ac.uk/~rja14/Papers/SEv3-ch19-7sep.pdf
2. https://www.intel.com/content/www/us/en/developer/articles/technical/software-security-guidance/best-practices/refined-speculative-execution-terminology.html
3. https://cwe.mitre.org/data/definitions/514.html

CWE-514: Covert Channel

https://cwe.mitre.org/data/definitions/514.html

Abstraction: Class

Description: A covert channel is a path that can be used to transfer information in a way not intended by the system's designers.

Extended Description: Typically the system has not given authorization for the transmission and has no knowledge of its occurrence.

Relationships:

ChildOf	CWE-1229:Creation of Emergent Resource
ParentOf	CWE-385:Covert Timing Channel
ParentOf	CWE-515: Covert Storage Channel
CanFollow	CWE-205: Observable Behavioral Discrepancy

Vulnerability Mapping Notes:

Usage: Allowed-with-Review; **Reason:** Abstraction

Rationale: This CWE entry is a Class and might have Base-level children that would be more appropriate; **Comments:** Examine children of this entry to see if there is a better fit.

NOTE: Nothing about EM based Covert Channels, nor HW cause, e.g., SMPS

Discussion

Questions:

- Should we place CWE-514 in the HW View?
- Do we need to modify CWE-514 to be less software centric?
- Or create a base of CWE-514 and put that into the HW view?

Previous HW SIG Member Comments:

- Covert Channels should have coverage in the hardware view *–Jason Oberg*
- Covert Channels should be in the HW categories Security Flow Issues, General Circuit and Logic Design Concerns, or Debug and Test Problems. *– Paul Wortman*

Most Important Hardware Weaknesses Refresh

Bob H

Most Important Hardware Weaknesses (MIHW)

- **Is this something worth revisiting?**
- **Part of CWE 4.6 Release, October 28, 2021**
- **Have there been substantial developments since the last release of MIHW?**
- **Would those affect the rankings and inclusions of the list in any meaningful way?**

Current MIHW

CWE-1189	Improper Isolation of Shared Resources on System-on-a-Chip (SoC)
CWE-1191	On-Chip Debug and Test Interface With Improper Access Control
CWE-1231	Improper Prevention of Lock Bit Modification
CWE-1233	Security-Sensitive Hardware Controls with Missing Lock Bit Protection
CWE-1240	Use of a Cryptographic Primitive with a Risky Implementation
CWE-1244	Internal Asset Exposed to Unsafe Debug Access Level or State
CWE-1256	Improper Restriction of Software Interfaces to Hardware Features
CWE-1260	Improper Handling of Overlap Between Protected Memory Ranges
CWE-1272	Sensitive Information Uncleared Before Debug/Power State Transition
CWE-1274	Improper Access Control for Volatile Memory Containing Boot Code
CWE-1277	Firmware Not Updateable
CWE-1300	Improper Protection of Physical Side Channels

New HW CWEs Since MIHW

- **CWE-1342: Information Exposure through Microarchitectural State after Transient Execution**
- **CWE-1357: Reliance on Insufficiently Trustworthy Component**
- **CWE-1384: Improper Handling of Physical or Environmental Conditions**
- **CWE-1388: Physical Access Issues and Concerns**

Discussion

- **Have there been substantial developments since the last release of MIHW?**
- **Would those affect the rankings and inclusions of the list in any meaningful way?**
- **Are there observational trends that would change the current list in any significant and meaningful way?**