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

April 11, 2025

Agenda

REMINDER: This meeting is being recorded.

1	General Status of Current HW CWE Submissions	Bob H	15 min
2	Most Important Hardware Weaknesses Refresh Update	Arun K, Ganu K	10 min

Housekeeping

- **Schedule:**
  - **Next Meeting May 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

- **CWE 4.17 has been released.**
  - 3 new weaknesses published (2 HW CWEs).
  - Usability improvements to 22 CWEs .
  - Updates to table formats (visually easier to consume).
  - More details here: CWE - News & Events – 2025.
- **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.

Call for Topics

What topics should we cover next time?

- **Anything to share today or topics for consideration for next meeting?**

Hardware Submissions in Progress

- **Initial Consultation (Stage 1, Phase 4 – common bottleneck)**
  - HW/SW: ES2306-c0b52346 Use of a Quantum-Vulnerable Cryptographic Algorithm (NIST)
- **Detailed Submission (Stage 2, Phase 7)**
  - ES2208-9fb81a1a Speculative propagation of requests for transaction before data validation in multi-manager bus architectures (Francesco Restuccia)
- **See CDR: https://github.com/CWE-CAPEC/CWE-Content-Development-Repository/issues**

CWE-1429: Missing Security-Relevant Feedback for Unexecuted Operations in Hardware Interface

Weakness ID: 1429
Vulnerability Mapping: ALLOWED
Abstraction: Base

Description

The product has a hardware interface that silently discards operations in situations for which feedback would be security-relevant, such as the timely detection of failures or attacks.

Extended Description

While some systems intentionally withhold feedback as a security measure, this approach must be strictly controlled to ensure it does not obscure operational failures that require prompt detection and remediation. Without these essential confirmations, failures go undetected, increasing the risk of data loss, security vulnerabilities, and overall system instability. Even when withholding feedback is an intentional part of a security policy designed, for example, to prevent attackers from gleaning sensitive internal details, the absence of expected feedback becomes a critical weakness when it masks operational failures that require prompt detection and remediation.

For instance, certain encryption algorithms always return ciphertext regardless of errors to prevent attackers from gaining insight into internal state details. However, if such an algorithm fails to generate the expected ciphertext and provides no error feedback, the system cannot distinguish between a legitimate output and a malfunction. This can lead to undetected cryptographic failures, potentially compromising data security and system reliability. Without proper notification, a critical failure might remain hidden, undermining both the reliability and security of the process.

Therefore, this weakness captures issues across various hardware interfaces where operations are discarded without any feedback, error handling, or logging. Such omissions can lead to data loss, security vulnerabilities, and system instability, with potential impacts ranging from minor to catastrophic.

For some kinds of hardware products, some errors may be correctly identified and subsequently discarded, and the lack of feedback may have been an intentional design decision. However, this could result in a weakness if system operators or other authorized entities are not provided feedback about security-critical operations or failures that could prevent the operators from detecting and responding to an attack.

For example:

- In a System-on-Chip (SoC) platform, write operations to reserved memory addresses might be correctly identified as invalid and subsequently discarded. However, if no feedback is provided to system operators, they may misinterpret the device's state, failing to recognize conditions that could lead to broader failures or security vulnerabilities. For example, if an attacker attempts unauthorized writes to protected regions, the system may silently discard these writes without alerting security mechanisms. This lack of feedback could obscure intrusion attempts or misconfigurations, increasing the risk of unnoticed system compromise
- Microcontroller Interrupt Systems: When interrupts are silently ignored due to priority conflicts or internal errors without notifying higher-level control, it becomes challenging to diagnose system failures or detect potential security breaches in a timely manner.
- Network Interface Controllers: Dropping packets - perhaps due to buffer overflows - without any error feedback can not only cause data loss but may also contribute to exploitable timing discrepancies that reveal sensitive internal processing details.

In HW View under “Cross-Cutting Problems”

CWE-1431: Driving Intermediate Cryptographic State/Results to Hardware Module Outputs

Weakness ID: 1431
Vulnerability Mapping: ALLOWED
Abstraction: Base

Description

The product uses a hardware module implementing a cryptographic algorithm that writes sensitive information about the intermediate state or results of its cryptographic operations via one of its output wires (typically the output port containing the final result).

Common Consequences

Impact	Details
Read Memory; Read Application Data	**Scope: Confidentiality** **Likelihood: Unknown** Mathematically sound cryptographic algorithms rely on their correct implementation for security. These assumptions might break when a hardware crypto module leaks intermediate encryption states or results such that they can be observed by an adversary. If intermediate state is observed, it might be possible for an attacker to identify the secrets used in the cryptographic operation.

Potential Mitigations

Phase(s)	Mitigation
Architecture and Design	Designers/developers should add or modify existing control flow logic along any data flow paths that connect "sources" (signals with intermediate cryptographic state/results) with "sinks" (hardware module outputs and other signals outside of trusted cryptographic zone). The control flow logic should only allow cryptographic results to be driven to "sinks" when appropriate conditions are satisfied (typically when the final result for a cryptographic operation has been generated). When the appropriate conditions are not satisfied (i.e., before or during a cryptographic operation), the control flow logic should drive a safe default value to "sinks". **Effectiveness: High**
Implementation	Designers/developers should add or modify existing control flow logic along any data flow paths that connect "sources" (signals with intermediate cryptographic state/results) with "sinks" (hardware module outputs and other signals outside of trusted cryptographic zone). The control flow logic should only allow cryptographic results to be driven to "sinks" when appropriate conditions are satisfied (typically when the final result for a cryptographic operation has been generated). When the appropriate conditions are not satisfied (i.e., before or during a cryptographic operation), the control flow logic should drive a safe default value to "sinks". **Effectiveness: High**

In HW View under “Security Primitives and Cryptography Issues”

Most Important Hardware Weaknesses List Update

Arun K., Gananand K.

Most Important Hardware Weaknesses Refresh – April 2025 Update

- **Finished vulnerability data collection and group is now analyzing that data.**

- **Schedule for Expert Opinion survey :**
  - Plan to provide above data analysis summary as a reference point for the expert opinion survey. (April 18, 2025)
  - Plan to conduct survey before next HW CWE SIG in two parts:
    - First part will ask HW CWE SIG members about what CWEs should be considered for inclusion in the updated MIHW list based on 9 significance questions. (April 23 – April 30)
    - Second part will ask HW CWE SIG members to rate the inclusion of specific HW CWEs based on a Likert scale. (May 2 – May 9)

- **MIHW Group to later combine the two data sets to derive the list**

Combining Vulnerability Data and Expert Opinion Data

- **Combine both vulnerability data and expert opinion, but how?**
  - Data & Expert Opinion
    - If both data and expert opinion point to a particular CWE, it should be included on the MIHW list.
  - No Data & Expert Opinion
    - If experts identify a CWE not highlighted by the data, but expert consensus is strong, include it with a note on expert-driven inclusion.
  - Data & No Expert Opinion
    - If data indicates a weakness but experts do not prioritize it, then it should not be included in the MIHW list.
  - No Data & No Expert Opinion
    - Exclude CWE from the MIHW list, as there is no supporting evidence or expert consensus.

MIHW List Discussion

- **Please continue discussion on the HW CWE Mailing List (see below).**
  - You can tag email subject lines using "[MIHW]" to get more visibility.

- **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/*

Next Meeting (May 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

Popular CWEs in NVD excluding Software CWEs

- **33 unique HW CWEs with *at least* 1 CVE in NVD.**
- **16 HW CWEs with *only* 1 CVE**
- **11 HW CWEs with 2 CVEs**

13	CWE-1220: Insufficient Granularity of Access Control
4	CWE-1320: Improper Protection for Outbound Error Messages and Alert Signals
3	CWE-1357: Reliance on Insufficiently Trustworthy Component
3	CWE-1295: Debug Messages Revealing Unnecessary Information
3	CWE-1258: Exposure of Sensitive System Information Due to Uncleared Debug Information
3	CWE-1240: Use of a Cryptographic Primitive with a Risky Implementation
2	CWE-1332: Improper Handling of Faults that Lead to Instruction Skips
2	CWE-1319: Improper Protection against Electromagnetic Fault Injection (EM-FI)
2	CWE-1303: Non-Transparent Sharing of Microarchitectural Resources
2	CWE-1299: Missing Protection Mechanism for Alternate Hardware Interface
2	CWE-1298: Hardware Logic Contains Race Conditions
2	CWE-1281: Sequence of Processor Instructions Leads to Unexpected Behavior
2	CWE-1279: Cryptographic Operations are run Before Supporting Units are Ready
2	CWE-1269: Product Released in Non-Release Configuration
2	CWE-1263: Improper Physical Access Control

Formation of Ad-Hoc Committee

- **Will be putting a call out of the mailing list for members to join an ad-hoc committee to study.**
- **We will be looking for committee members to study the feasibility of a new list and making a decision to proceed.**
- **Also, members will develop an approach to develop the list with the community.**

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?
- Is there any data available that we could utilize to generate the list? Or should we use the delphi method again?
- Are there observational trends that would change the current list in any significant and meaningful way?

Current MIHW from CWE 4.6 (Unranked)

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

Previous Methodology for the MIHW

- **Previously, the Delphi method was used to poll members of this august body:**
  - Which 10 HW weaknesses were important based on nine significance questions:
    1. How frequently is this weakness detected after it has been fielded?
    2. Does the weakness require hardware modifications to mitigate it?
    3. How frequently is this weakness detected during design?
    4. How frequently is this weakness detected during test?
    5. Can the weakness be mitigated once the device has been fielded?
    6. Is physical access required to exploit this weakness?
    7. Can an attack exploiting this weakness be conducted entirely via software?
    8. Is a single exploit against this weakness applicable to a wide range (or family) of devices?
    9. What methodologies do you practice for identifying and preventing both known weaknesses and new weaknesses?

Previous Methodology for MIHW

- **After combining the above, thirty-one unique weaknesses resulted.**
- **Live poll conducted during SIG meeting asked members to assign the thirty-one into various buckets with weights (strongly support (+2), somewhat support (+1), no opinion (0), somewhat oppose (-1), strongly oppose (-2)).**
- **Multiple groups emerged from the data when ranked by weighted percentage of votes, of which the primary group contained twelve. The secondary group were listed as Hardware Weaknesses on the Cusp (shown below).**

CWE-226	Sensitive Information in Resource Not Removed Before Reuse
CWE-1247	Improper Protection Against Voltage and Clock Glitches
CWE-1262	Improper Access Control for Register Interface
CWE-1331	Improper Isolation of Shared Resources in Network On Chip (NoC)
CWE-1332	Improper Handling of Faults that Lead to Instruction Skips

Changes to HW CWEs since MIHW (v4.6 to v4.16)

- (DEPRECATED) CWE-1324: Sensitive Information Accessible by Physical Probing of JTAG Interface
- CWE-1342: Information Exposure through Microarchitectural State after Transient Execution
- (Class) CWE-1357: Reliance on Insufficiently Trustworthy Component
  - CWE-1329: Reliance on Component That is Not Updateable
- (Class) CWE-1384: Improper Handling of Physical or Environmental Conditions
- (Category) CWE-1388: Physical Access Issues and Concerns
- (Parent) CWE-1420: Exposure of Sensitive Information during Transient Execution
  - CWE-1421: Exposure of Sensitive Information in Shared Microarchitectural Structures during Transient Execution
  - CWE-1422: Exposure of Sensitive Information caused by Incorrect Data Forwarding during Transient Execution
  - CWE-1423: Exposure of Sensitive Information caused by Shared Microarchitectural Predictor State that Influences Transient Execution
- This list is not complete!

Questions to think about and frame the MIHW activity

- **Unfortunately, not many HW CWEs assigned to CVE entries in NVD currently.**
- **A lot of the observed examples (CVE) were added mostly for Transient Execution weaknesses. There are many that remain incomplete.**
- **What methodology should be used for the new MIHW list?**
- **What data could be used to generate the new MIHW list?**
- **Any observations or lessons learned that could change or impact the current list in a meaningful way? Any observed trends that could impact the list?**
- **Does it capture the use cases and important attributes discussed earlier for such a list?**