NIST Internal Report
NIST IR 8517 ipd

Hardware Security Failure Scenarios

Potential Weaknesses in Hardware Design

Initial Public Draft

Peter Mell
Irena Bojanova

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8517.ipd

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
U.S. DEPARTMENT OF COMMERCE

NIST Internal Report
NIST IR 8517 ipd

Hardware Security Failure Scenarios

Potential Weaknesses in Hardware Design

Initial Public Draft

Peter Mell
Computer Security Division
Information Technology Laboratory

Irena Bojanova
Software and Systems Division
Information Technology Laboratory

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8517.ipd

June 2024

U.S. Department of Commerce
Gina M. Raimondo, Secretary

National Institute of Standards and Technology
Laurie E. Locascio, NIST Director and Under Secretary of Commerce for Standards and Technology

Publication History
Approved by the NIST Editorial Review Board on YYYY-MM-DD [will be added in final publication]

How to Cite this NIST Technical Series Publication
Mell P, Bojanova I (2024) Hardware Security Failure Scenarios: Potential Weaknesses in Hardware Design. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Interagency or Internal Report (IR) NIST IR 8517 ipd. https://doi.org/10.6028/NIST.IR.8517.ipd

Author ORCID iDs
Peter Mell: 0000-0003-2938-897X
Irena Bojanova: 0000-0002-3198-7026

Contact Information
nistir8517@nist.gov

National Institute of Standards and Technology
Attn: Computer Security Division, Information Technology Laboratory
100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930

Public Comment Period
June 13, 2024 – July 31, 2024

Submit Comments
nistir8517@nist.gov

National Institute of Standards and Technology
Attn: Computer Security Division, Information Technology Laboratory
100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930

Additional Information
Additional information about this publication is available at https://csrc.nist.gov/pubs/ir/8517/ipd, including related content, potential updates, and document history.

All comments are subject to release under the Freedom of Information Act (FOIA).

Abstract

Historically, hardware has been assumed to be inherently secure. However, chips are both created with and contain complex software, and software is known to have bugs. Some of these bugs will compromise security. This publication evaluates the types of vulnerabilities that can occur, leveraging existing work on hardware weaknesses. For each type, a security failure scenario is provided that describes **how** the weakness could be exploited, **where** the weakness typically occurs, and **what** kind of damage could be done by an attacker. The 98 failure scenarios provided demonstrate the extensive and broadly distributed possibilities for hardware-related security failures.

Keywords

chips; design; failures; hardware; scenarios; security; vulnerability; weakness.

Reports on Computer Systems Technology

Audience

This report is intended for a broad audience who wants to understand the many ways in which hardware can fail from a security perspective. This includes policymakers interested in information technology (IT) security, IT security officers, operation security staff who must secure deployed hardware, and developers of hardware. It is written for a technically oriented audience, but it does not require specific knowledge of hardware security.

Table of Contents

List of Figures

1. Introduction

Historically, hardware has been viewed as “an immutable root-of-trust” with no security issues [1]. It has been assumed to be inherently secure. However, chips are created with and contain complex software, and software is known to have bugs. It is not unusual to have 1-25 bugs per 1000 lines of code for delivered software [2], and some of these bugs will have security implications. Further complicating matters, many of these bugs are hard-coded onto silicon, which can make mitigation challenging.

This work describes and categorizes ways in which computer hardware (HW) (i.e., chips) can fail from a security perspective. It does this by enumerating 98 scenarios that represent potential weaknesses in the programming and physical aspects of HW design. The purpose is to highlight the dangers of vulnerabilities potentially being introduced into the HW design process.

The Common Weakness Enumeration (CWE) [8][9] is a list of weaknesses. In this context, a weakness is defined as “a condition in a software, firmware, hardware, or service component that, under certain circumstances, could contribute to the introduction of vulnerabilities” [4]. CWE designators of the form (CWE-XXXX) are given to each of the 934 listed weaknesses (as of January 26, 2024). Each weakness entry contains complex, multi-page data elements with detailed security information. Since the inception of CWEs, a primary focus has been software weaknesses, while coverage of hardware-specific weaknesses has been more recent. All CWEs can be viewed by using the ‘ID Lookup’ search box on the CWE webpage [9].

As of April 29, 2024, the HW CWE Special Interest Group (HW CWE SIG) [5] has curated a list of 108 HW CWEs focused on HW design issues. The list includes a few CWEs that were created for software weaknesses but that are also relevant to HW weaknesses. These ‘software’ CWEs have been expanded to include HW-specific details and examples. However, the majority of the CWEs on the list are HW-specific and do not apply to the software domain. This indicates that HW security is fundamentally different from software security, despite the fact that both are created with and contain code. This publication demonstrates the uniqueness of HW security and the very different challenges presented compared to software security. At the same time, HW can contain weaknesses commonly found in software, and an HW weakness may be linked in a chain of weaknesses that include software weaknesses.

The HW security failure scenarios in this publication are based on the HW CWEs. For the purposes of this publication, an HW security failure scenario briefly describes how an attacker can cause a particular type of damage where the exploit typically occurs. Focusing on weaknesses enables one to look at the set of potential dangers, inclusive of and beyond the set of publicly published vulnerabilities. While reasonably comprehensive, the failure scenarios are not intended to provide exhaustive coverage. Their purpose is to highlight the significant danger presented by each HW weakness.

2. Background

This section provides the background for understanding the technical approach and categorization system used in creating and organizing the HW security failure scenarios. Readers interested in simply perusing the failure scenarios without understanding how they were derived or organized should go directly to Sec. 4.

2.1. Weaknesses vs. Vulnerabilities

A weakness can also be defined as a bug or fault type that can be exploited through an operation that results in a security-relevant error [3]. The word ‘type’ is critical as it conveys that a weakness is a concept that can be instantiated in software or hardware; a weakness is not specific to a particular program or chip. A vulnerability, however, is tied to a specific piece of code or chip. A vulnerability is an instantiation of a weakness. Complicating matters, some vulnerabilities arise only in the context of a chain of weaknesses [3].

Vulnerabilities are enumerated in the Common Vulnerabilities and Exposures (CVE) list [6]. The National Vulnerability Database contains details on each CVE [7]. There are over 25,000 CVEs published annually, with the rate usually growing each year. As of February 22, 2024 only 131 of these are HW CVEs.

2.2. Weakness Data Fields

Every weakness in the CWE is described by a set of elements. The following are the CWE data fields leveraged in the creation of the HW failure scenarios:

1. **Description/Extended Description —** Detailed explanation of the fault type
2. **Relationships/Memberships —** Taxonomic information to organize weaknesses into hierarchies and categories
3. **Modes of Introduction —** Descriptions of the life cycle phase where the CWE can be introduced
4. **Applicable Platforms —** Involved languages and technologies
5. **Common Consequences —** Affected security attributes along with likelihoods (e.g., confidentiality, integrity, availability, access control, authentication, and authorization)
6. **Demonstrative Examples —** Hypothetical examples of the weakness
7. **Observed Examples —**Actual observed examples of the weakness, usually with CVE references
8. **Potential Mitigations —** Protection methods

2.3. Weakness Abstractions

The CWE weaknesses model is composed of four layers of abstraction: pillar (P), class (C), base (B), and variant (V)[1]. The abstraction reflects the extent to which issues are being described in terms of five dimensions: behavior, property, technology, language, and resource. Variant weaknesses are at the most specific level of abstraction and describe at least three dimensions. Base weaknesses are more abstract than variants and more specific than classes; they describe two to three dimensions. Class weaknesses are very abstract and not typically specific about any language or technology; they describe one to two dimensions. Pillar weaknesses are at the highest level of abstraction. In this work, pillars and classes are used to organize the HW security failure scenarios.

2.4. Weakness Views

CWE designators of the form (CWE-XXXX) are given to weaknesses, views, and categories. A view provides a hierarchical organization of CWEs from a particular perspective (e.g., software development, research, and hardware design). A category is a simpler construct that groups a set of CWEs that have some similarity. Views may contain categories within their hierarchy.

As of February 9, 2024, the CWE contains 49 views and 374 categories. There are three views pertinent to this work: Hardware Design view (CWE-1194), Research Concepts view (CWE-1000), and the Weaknesses for Simplified Mapping of Published Vulnerabilities view (CWE-1003).

2.4.1. Hardware Design View

The Hardware Design view (CWE-1194) organizes the 108 HW weakness CWEs using 13 categories. This view is a three-level hierarchy with CWE-1194 as its root, the 13 categories[2] as children of the root, and a tree of HW weakness CWEs under each category. HW weaknesses may occur under multiple categories, although most do not.

The 13 categories of HW design weaknesses are:

1. Core and Compute Issues (CWE-1201)
2. Cross-Cutting Problems (CWE-1208)
3. Debug and Test Problems (CWE-1207)
4. General Circuit and Logic Design Concerns (CWE-1199)
5. Integration Issues (CWE-1197)
6. Manufacturing and Life Cycle Management Concerns (CWE-1195)
7. Memory and Storage Issues (CWE-1202)

[1] A compound element (linking together weaknesses) associates two or more interacting or co-occurring CWEs. None of the HW CWEs are of the compound abstraction.
[2] Section 5 provides details on the 13 categories.

8. Peripherals, On-chip Fabric, and Interface/IO Problems (CWE-1203)
9. Physical Access Issues and Concerns (CWE-1388)
10. Power, Clock, Thermal, and Reset Concerns (CWE-1206)
11. Privilege Separation and Access Control Issues (CWE-1198)
12. Security Flow Issues (CWE-1196)
13. Security Primitives and Cryptography Issues (CWE-1205)

2.4.2. Research Concepts View

The Research Concepts view (CWE-1000) organizes all weakness CWEs by the method through which an exploitation can occur. It is a directed acyclic graph with a single source node, CWE-1000. In this hierarchy, some CWEs can have multiple parents, and all of them have CWE-1000 as their oldest ancestor. These properties allow a CWE (even one with only one parent) to possibly be reached through multiple paths from the root.

The children of CWE-1000 are 10 pillars that organize the weakness CWEs. The pillar CWEs marked with * contain HW CWEs. However, none of these pillars are hardware-specific and cover many software security weaknesses as well.

1. Improper Access Control (CWE-284) *
2. Improper Adherence to Coding Standards (CWE-710) *
3. Improper Check or Handling of Exceptional Conditions (CWE-703) *
4. Improper Control of a Resource Through its Lifetime (CWE-664) *
5. Improper Interaction Between Multiple Correctly-Behaving Entities (CWE-435)
6. Improper Neutralization (CWE-707)
7. Incorrect Calculation (CWE-682)
8. Incorrect Comparison (CWE-697)*
9. Insufficient Control Flow Management (CWE-691) *
10. Protection Mechanism Failure (CWE-693) *

2.4.3. Simplified Mapping of Published Vulnerabilities View

The Weaknesses for Simplified Mapping of Published Vulnerabilities view (CWE-1003) organizes the weaknesses that are most commonly seen in software CVEs to assist organizations that deal with such data (e.g., vulnerability databases and security tool vendors).

It is a three-level tree with CWE-1003 as its root (i.e., there is only one path to each CWE, and all CWEs have exactly one parent). It has no categories and organizes the CWEs by pillars and classes. The children of the root are 35 classes and two pillars. It contains a total of 130 weaknesses, and only three of these weaknesses are also HW CWEs (CWE-203, CWE-276, and CWE-319).

3. Technical Approach

This section describes the concept of a hardware security failure scenario and the approach to creating weakness graphs to organize them.

3.1. Concept of Hardware Security Failure Scenarios

For the purposes of this work, a hardware security failure scenario describes a malicious entity (e.g., human attacker or automated malware) leveraging a weakness to violate security policy. Each failure scenario has three aspects: **how** the weakness could be exploited, **where** the weakness typically occurs, and **what** kind of damage could be done.

While reasonably comprehensive, the failure scenarios are not intended to provide exhaustive coverage. Their purpose is to highlight the dangers presented by each HW weakness.

3.1.1. Determining How Weaknesses Occur

The ‘Extended Description’ and ‘Modes of Introduction’ sections of each CWE entry provide information on how an HW CWE can occur. The CWE Research Concepts view (CWE-1000) organizes HW CWEs by abstractions of behavior. The path of nodes from the Research Concepts view root to the HW CWE under analysis describes how a weakness can occur with increasing granularity as the path is traversed. Some HW CWEs have multiple paths that typically describe simultaneously occurring behaviors and provide a more complete picture of how these CWEs occur.

3.1.2. Determining Where Weaknesses Occur

The Hardware Design view (CWE-1194) organizes the HW CWEs into 13 categories. They generally describe where an HW CWE can occur, potentially from different points of view (e.g., physically on the chip, security operations, and life cycle). Section 5 describes each of these categories and the CWE classes associated with them. The ‘Extended Description’ of each CWE is usually helpful in determining the “where.”

3.1.3. Determining What Damage Weaknesses Allow

The CWE entry ‘Common Consequences’ section provides a high-level list of the security areas affected (e.g., access control, confidentiality, integrity, and availability) and the technical impacts (e.g., read data, modify data, bypass access control). The ‘Observed Examples’ section provides more granular and concrete damage explanations that are often useful for creating failure scenarios. The ‘Extended Description’ section often discusses potential damage.

3.2. Creating Hardware Weakness Subgraphs

The failure scenarios are organized by their associated HW CWEs. The HW CWEs are primarily organized by the Research Concepts view (CWE-1000) and then secondarily by the Hardware view ([CWE-1194](CWE-1194)). This approach provides directed graphs that hierarchically show **how** HW CWEs occur at increasing levels of granularity as the graph is traversed and additional information is added about **where** the weaknesses can occur.

Figure 1 shows the complete HW CWE graph, all of the HW CWEs, and the non-HW CWEs necessary to connect them together.

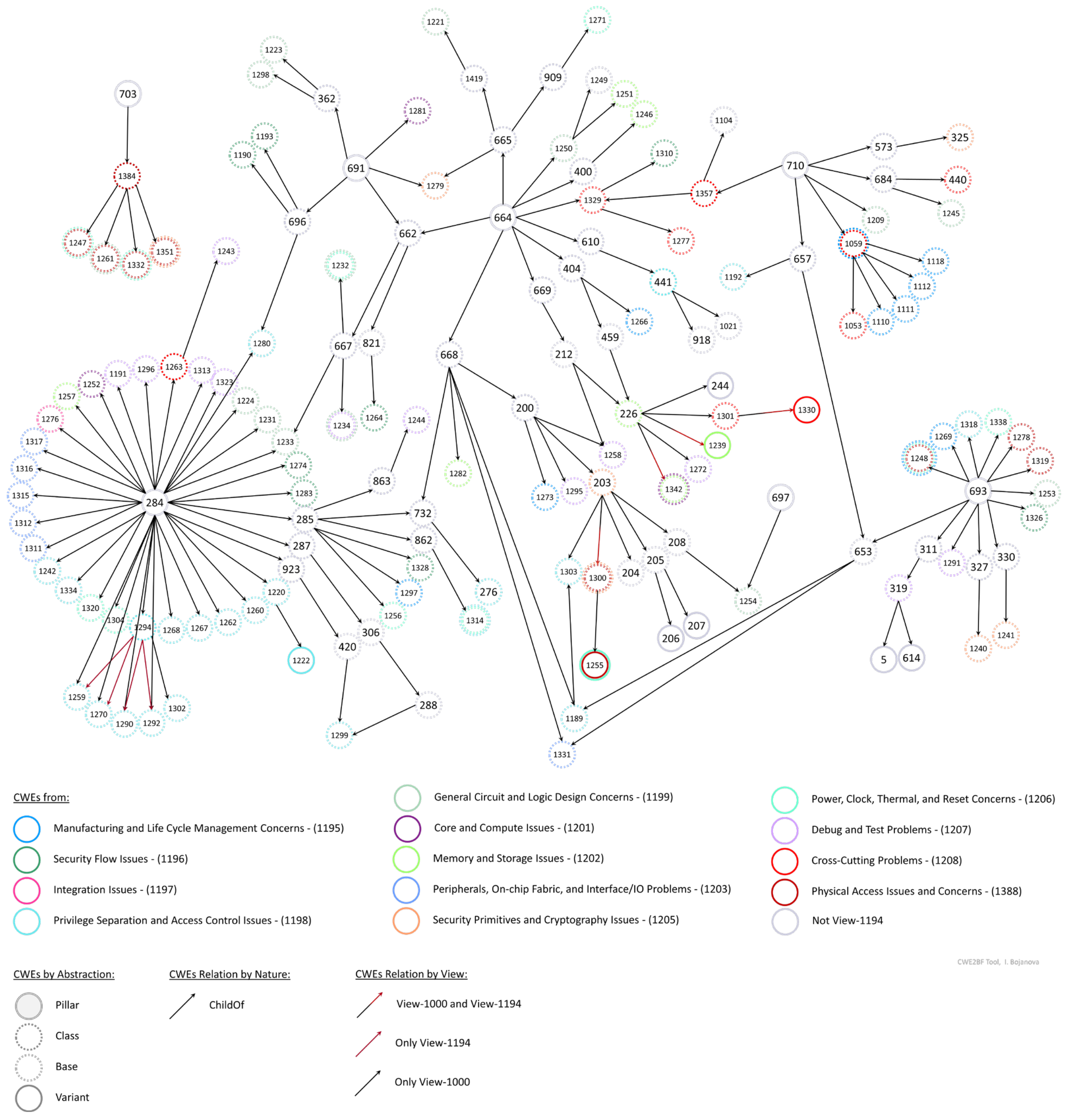


Fig. 1. Complete HW CWE graph created using View 1000 and View 1194

The HW CWE graph contains a root node for each of the seven Research Concepts view ([CWE-1000)](CWE-1000) pillars that contain HW CWEs. It shows the Hardware Design view ([CWE-1194](CWE-1194)) categories to which each CWE belongs and the view from which each relationship was defined. It also shows the abstraction for each CWE pillar, class, base, and variant.

Section 4 shows the subgraphs of the CWEs reachable from each respective HW-associated pillar. Appendix B provides an analysis and statistics for Fig. 1 and describes the algorithm used for the construction of the graphs. Appendix C through Appendix I provide an alternative textual view of the pillar subtrees using a strict hierarchical tree layout. This latter approach is convenient for a quick perusal of the HW CWEs but cannot capture the complex relationships that only become apparent from the complete graph view.

The HW CWE graphs in this publication primarily use arrows to show the relationships between the CWEs and colors to quickly provide additional information about each CWE (e.g., the HW category it belongs to and the abstraction). For readers with difficulties discerning the colors, this same information is available for each CWE on the associated CWE web page and can be accessed using the format https://cwe.mitre.org/data/definitions/XXX.html, where XXX is replaced with the CWE number.

4. Hardware Security Failure Scenarios

The HW security failure scenarios were created by reviewing the full CWE entries, extracting the three failure scenario aspects (the 'how', 'when', and 'what' from Sec. 3.1), and then writing a short summary of those aspects.

This section contains an enumeration of 98 HW security failure scenarios distributed among the CWE pillars as follows:

1. Improper Access Control (CWE-284, 43 scenarios)
2. Improper Adherence to Coding Standards (CWE-710, 14 scenarios)
3. Improper Check or Handling of Exceptional Conditions (CWE-703, five scenarios)
4. Improper Control of a Resource Through its Lifetime (CWE-664, 40 scenarios)
5. Incorrect Comparison (CWE-697, one scenario)
6. Insufficient Control Flow Management (CWE-691, 11 scenarios)
7. Protection Mechanism Failure (CWE-693, 15 scenarios)

The presence of a failure scenario in a product indicates the presence of the associated weakness and an issue with one of the above pillars.

A small number of HW CWEs fall under multiple pillars. For these CWEs, the associated security failure scenario is located in the section for the pillar that qualitatively has the strongest linkage to the CWE. The full CWE Research Concepts view graph in Appendix B shows which HW CWEs are shared under which pillars.

The HW CWEs are grouped by the classes underlying the pillar. The CWE Research Concepts view often provides finer grained delineations (e.g., organizing bases and variants under other bases or providing subclasses under classes). For clarity of reading, this additional information is provided in the associated figures for each subsection with directed subgraphs of the HW CWEs under each pillar.

4.1. Improper Access Control

The CWE Improper Access Control (CWE-284) applies when a "product does not restrict or incorrectly restricts access to a resource from an unauthorized actor." Access control involves the use of protection mechanisms, such as:

- Authentication (i.e., proving the identity of an actor)
- Authorization (i.e., ensuring that a given actor can access a resource)
- Accountability (i.e., tracking activities that were performed)

The HW CWEs under this pillar occur within the following pillar/class hierarchy. The CWEs marked with * are HW CWEs.

CWE-284 P Improper Access Control

- CWE-1263 C Improper Physical Access Control *
- CWE-1294 C Insecure Security Identifier Mechanism *
- CWE-285 C Improper Authorization

Figure 2 shows the directed graph of HW CWEs under this pillar with their parent-child relationships.

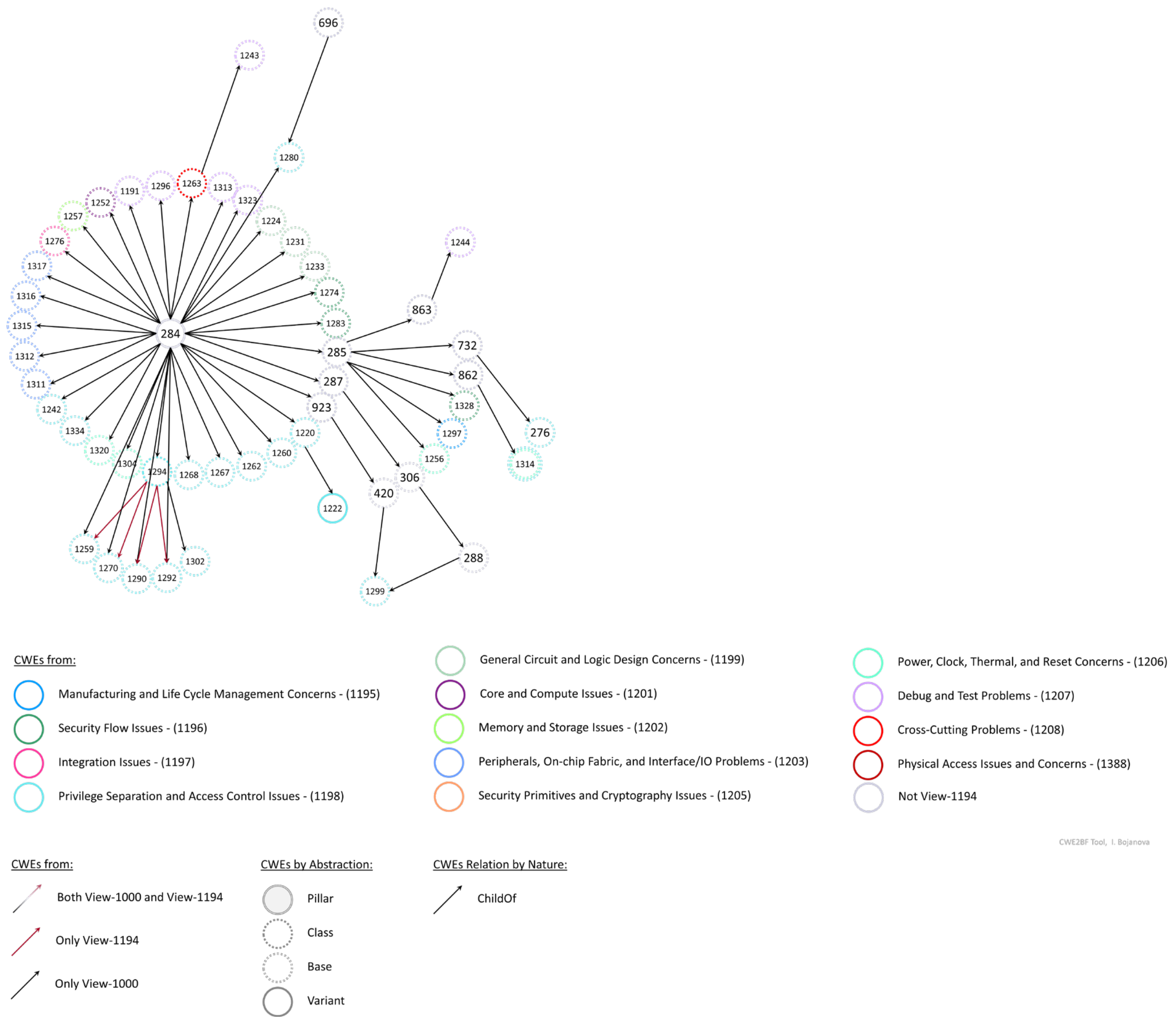

696
1243
1280
1263
1296
1313
1191
1323
1252
1257
1224
1276
1231
1317
1233
1316
1274
1315
284
1283
863
1244
1312
285
732
1311
287
862
1242
923
1328
1334
1220
1297
276
1320
1260
1256
1314
1304
1294
1268
1267
1262
306
420
1222
288
1259
1270
1290
1292
1302
1299
CWEs from:
Manufacturing and Life Cycle Management Concerns - (1195)
Security Flow Issues - (1196)
Integration Issues - (1197)
Privilege Separation and Access Control Issues - (1198)
General Circuit and Logic Design Concerns - (1199)
Core and Compute Issues - (1201)
Memory and Storage Issues - (1202)
Peripherals, On-chip Fabric, and Interface/IO Problems - (1203)
Security Primitives and Cryptography Issues - (1205)
Power, Clock, Thermal, and Reset Concerns - (1206)
Debug and Test Problems - (1207)
Cross-Cutting Problems - (1208)
Physical Access Issues and Concerns - (1388)
Not View-1194
CWE2BF Tool, I. Bojanova
CWEs from:
Both View-1000 and View-1194
Only View-1194
Only View-1000
CWEs by Abstraction:
Pillar
Class
Base
Variant
CWEs Relation by Nature:
ChildOf

Fig. 2. HW CWE subgraph for pillar Improper Access Control (CWE-284)

The HW class Improper Physical Access Control (CWE-1263) has one HW CWE child (CWE-1243). The security failure scenario is:

1. A malicious human can leverage physical access to obtain restricted information because the physical security features are insufficient [CWE-1263].
    a. During debug operations, an untrusted agent can read security-sensitive device information (e.g., encryption keys and manufacting data) that is permanently

stored in fuses but loaded into protected registers due to code that does not take the debug mode into account [CWE-1243].

The HW class Insecure Security Identifier Mechanism (CWE-1294) has five HW CWE children. The security failure scenarios are:

1. A malicious agent can initiate an unauthorized transaction (e.g., read, write, program, reset, fetch, compute) by taking advantage of incorrectly implemented security identifiers that define the privilege level of the agent in a system-on-a-chip (SoC) [CWE-1294].
    a. A malicious agent on an SOC may assign itself inappropriate security tokens to give itself additional privileges (e.g., read, write, fetch, program, compute, reset) because the security tokens are improperly protected [CWE-1259].
    b. A malicious agent can gain inappropriate privileges over assets due to an incorrect assignment of security tokens to agents. A single token may be assigned to multiple agents, or multiple tokens may be assigned to a single agent [CWE-1270].
    c. A malicious agent can gain unauthorized access to an asset by taking advantage of the incorrect decoding of security identifier information in bus-transaction signals [CWE-1290].
    d. An agent can gain unauthorized access to an asset by taking advantage of a bridge incorrectly performing a protocol conversion between agents that use different bus protocols [CWE-1292].
    e. A security identifier is not included with an agent-to-agent transaction. This can result in a denial of service (DoS) for the agent's requests or the ability of a malicious agent to enact unauthorized actions due to inappropriate handling of the missing identifier by the destination agent [CWE-1302].

The non-HW class Improper Authorization (CWE-285) has five security failure scenarios:

1. Malicious software can take advantage of software-controllable device functionality (e.g., power control, clock management, and memory access) to modify registers/memory or to perform side-channel attacks without the need for physical access to the chip [CWE-1256].
2. A malicious actor at an outsourced semiconductor assembly and test (OSAT) facility can take advantage of logic errors in debug interconnections to obtain improper access to sensitive information for chips in the more vulnerable pre-production stage [CWE-1297].
3. An attacker can modify the hardware-stored firmware version number used in the secure or verified boot process. The attacker can then execute older vulnerable versions of firmware with plans to exploit known vulnerabilities and possibly prevent upgrades [CWE-1328].
4. Malicious software can change non-write-protected parametric data values, thus changing the unit conversion/scaling for sensor reporting (e.g., thermal, power, voltage,

current, and frequency). This can cause hardware to operate outside of design limits even though the limit values themselves have not been modified [CWE-1314].

5. A human can use a physical debug or test interface to obtain sensitive information from an asset due to an incorrect debug access level assignment [CWE-1244].

There are 27 non-class HW CWEs that are direct children of pillar Improper Access Control (CWE-284). The security failure scenarios are:

1. An attacker with physical access to a chip can leverage a lack of or faults in debug/test interface access control to read and set registers (e.g., via a scan chain using a Joint Test Action Group [JTAG] interface) and bypass normal on-chip protections [CWE-1191].
2. Malicious code on a device may leverage a lack of granularity in hardware access control to read or modify assets (e.g., device configuration and keys) by taking advantage of unintended privileges [CWE-1220].
3. New functionality may not be implementable because a programmable lock bit set during the boot process prevents an unnecessarily large address region from being written [CWE-1222].
4. Malicious code can take advantage of an improper implementation of write-once register bits to reprogram system settings (e.g., boot time configuration) [CWE-1224].
5. Attackers may unlock a secured system by leveraging design or code errors to modify trusted lock bits that should have their values immutable after the initial set, thereby enabling writes to protected registers or address regions [CWE-1231].
6. Attackers can gain full read-write access to a device by accessing undocumented features (typically put there to allow for easy developer testing) that circumvent security controls. These are often implemented as “chicken bits” — undocumented bits that disable security features [CWE-1242].
7. Attackers can write malicious code to memory and then execute it because the central processing unit (CPU) does not support a bit that defines read-only and write-only regions of memory. This can also happen if the CPU relies on an improperly configured memory protection unit (MPU) and memory management unit (MMU) for read and write exclusivity [CWE-1252].
8. Attackers can access protected memory regions and perform both read and write by using memory alias addresses (i.e., redundant addresses that point to the same memory region) or mirrored memory regions that do not have the same protections. An attacker could possibly create memory address aliases to perform such an attack [CWE-1257].
9. Lower privilege software can write to memory regions for higher privileged software due to overlapping memory regions, thus enabling malicious software to perform privilege escalation or a DoS attack [CWE-1260].
10. Malicious software can access registers that provide hardware functionality interfaces due to an access control fault, allowing confidentiality and integrity violations [CWE-1262].

11. A malicious agent on an SoC may gain inappropriate or even full access to another agent when sending a bus transaction because the policy encoder mapping bus transactions to security tokens uses an obsolete encoding [CWE-1267].
12. A malicious agent can take advantage of improperly granted hardware control policy privileges to grant themselves read or write privileges over a protected resource (e.g., register-stored encryption keys) [CWE-1268].
13. An attacker can change or replace boot loader code by leveraging inadequate access control for the volatile memory (VM) in which the code is copied. This code is copied from non-volatile memory (NVM) to VM and then authenticated by the SoC read-only memory (ROM) code, but it is vulnerable to change after this occurs [CWE-1274].
14. Hardware intellectual property (IP) — an independently developed component — may be improperly connected to its parent and result in security risks due to incorrectly connected signaling. Functionality may be maintained but security weakened, enabling unauthorized access by external agents [CWE-1276].
15. Malicious code can modify the registers containing the attestation data that measures the boot code (i.e., secure hashes of the boot code), thereby enabling altered boot code to be executed without being detected [CWE-1283].
16. A human can obtain unauthorized access permissions through a test access port (TAP) or similar design element by leveraging logic errors that misconfigure the interconnections of debug components [CWE-1296].
17. When a product is powering down, an attacker can modify the configuration state being saved to persistent storage to alter the security or safety configuration upon restart (e.g., modify privileges, disable protections, or damage hardware) [CWE-1304].
18. Malicious software can bypass access controls by leveraging a bridge between IP blocks that use different fabric protocols (i.e., interconnecting components) that is incorrectly translating security attributes from one protocol to another [CWE-1311].
19. An attacker can bypass a firewall in an on-chip fabric by writing to an unprotected mirrored memory region that then propagates the changes to the original data [CWE-1312].
20. An attacker can leverage a hardware feature that allows for the activation of test or debug logic at runtime, thus enabling unauthorized reads and modifications to system data and bus messages [CWE-1313].
21. A malicious IP responder in a fabric may initiate control transactions to other devices through an incorrectly set register bit that allows an IP block to access other peripherals [CWE-1315].
22. Protected and unprotected memory regions for an on-chip fabric may have overlapping mappings (either accidentally or intentionally and maliciously) that enable an attacker to send a transaction that modifies protected memory [CWE-1316].

23. An attacker can gain unauthorized access to an IP block by leveraging a lack of access control checks by a fabric bridge that is translating transactions between two different protocols [CWE-1317].
24. A malicious agent can cause hardware to operate outside of its design limits (potentially causing physical damage) by disabling sensor alerts or initiate a DoS attack by generating alerts. The attacker may also disrupt the response mechanism that receives the alerts [CWE-1320].
25. An attacker can read security-sensitive traces (i.e., log data of IP blocks) from trace aggregation IP blocks that either store this data in unprotected memory or allow transport to unprivileged users (e.g., via a debug-trace port). These traces can include instructions executed from a CPU, transaction types and destinations from a fabric, and cryptographic keys from cryptographic coprocessors [CWE-1323].
26. An attacker can make unauthorized use of hardware error injection capabilities (normally used for testing) to disrupt redundant IP blocks, thereby degrading redundancy or forcing the IP component into a degraded operational mode [CWE-1334].
27. An attacker can bypass access control-protected assets by using unprotected alternate paths (e.g., shadow registers and external interfaces) [CWE-1299].

4.2. Improper Adherence to Coding Standards

The CWE Improper Adherence to Coding Standards (CWE-710) applies when a "product does not follow certain coding rules for development, which can lead to resultant weaknesses or increase the severity of the associated vulnerabilities."

The HW CWEs under this pillar occur within the following pillar/class hierarchy. The CWEs marked with * are HW CWEs.

CWE-710 P Improper Adherence to Coding Standards

- CWE-573 C Improper Following of Specification by Caller
- CWE-684 C Incorrect Provision of Specified Functionality
- CWE-1059 C Insufficient Technical Documentation *
- CWE-1357 C Reliance on Insufficiently Trustworthy Component *
- CWE-657 C Violation of Secure Design Principles

Figure 3 shows the directed graph of HW CWEs under this pillar with their parent-child relationships.

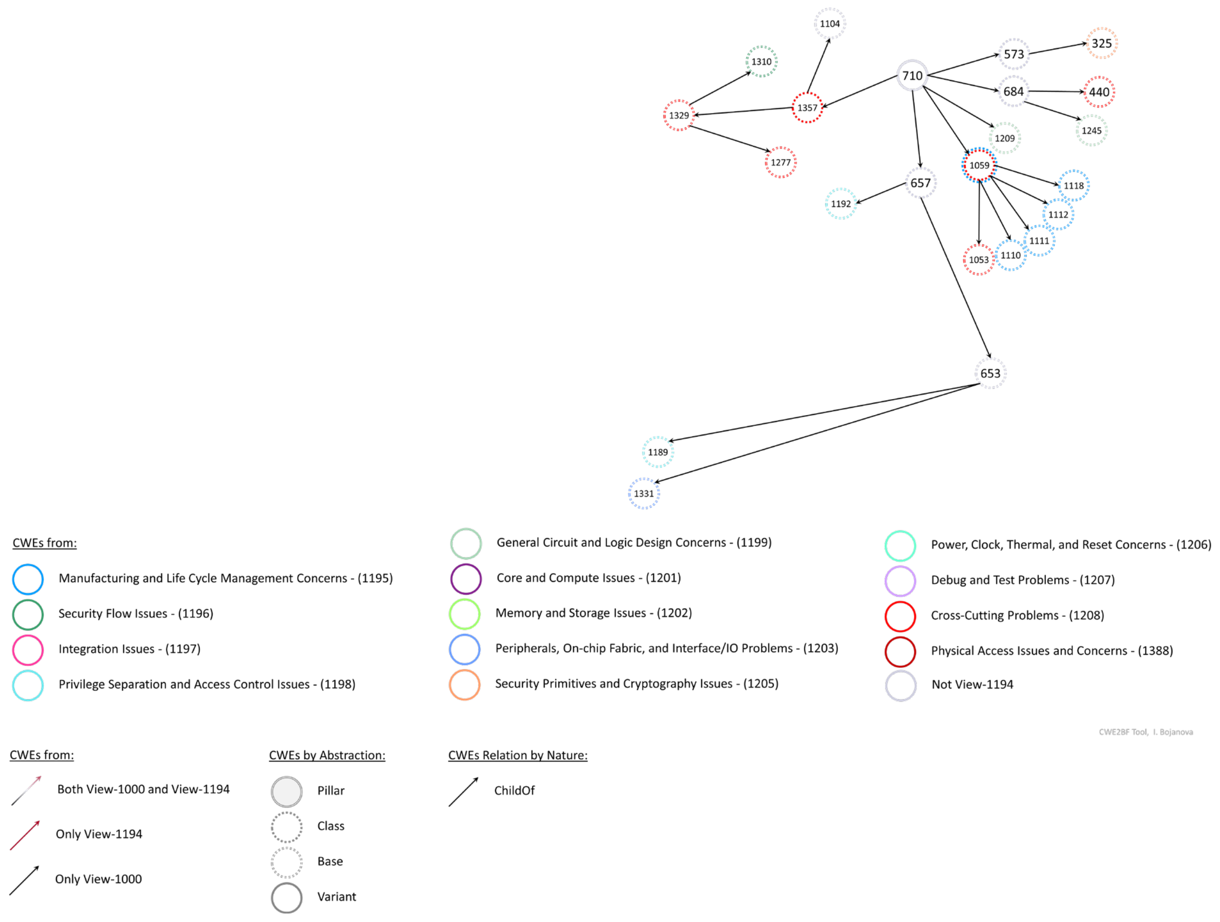

1104
1310
1329
1357
1277
710
573
325
684
440
1245
1209
1059
1118
1112
1111
1110
1053
657
1192
653
1189
1331
CWEs from:
Manufacturing and Life Cycle Management Concerns - (1195)
Security Flow Issues - (1196)
Integration Issues - (1197)
Privilege Separation and Access Control Issues - (1198)
General Circuit and Logic Design Concerns - (1199)
Core and Compute Issues - (1201)
Memory and Storage Issues - (1202)
Peripherals, On-chip Fabric, and Interface/IO Problems - (1203)
Security Primitives and Cryptography Issues - (1205)
Power, Clock, Thermal, and Reset Concerns - (1206)
Debug and Test Problems - (1207)
Cross-Cutting Problems - (1208)
Physical Access Issues and Concerns - (1388)
Not View-1194
CWE2BF Tool, I. Bojanova
CWEs from:
Both View-1000 and View-1194
Only View-1194
Only View-1000
CWEs by Abstraction:
Pillar
Class
Base
Variant
CWEs Relation by Nature:
ChildOf

Fig. 3. HW CWE subgraph for pillar Improper Adherence to Coding Standards (CWE-710)

Under the non-HW class Improper Following of Specification by Caller (CWE-573), there is one security failure scenario:

1. An attacker can decipher cryptographic output because the cryptographic algorithm used by the IP block does not implement a required step [CWE-325].

Under the non-HW class Incorrect Provision of Specified Functionality (CWE-684), there are two security failure scenarios:

1. An attacker can compromise security due to an IP block that fails to perform according to its specification [CWE-440].
2. Attackers can cause a DoS or possibly gain privileges by providing input to a finite state machine (FSM) that drives it in an undefined state (the FSM code does not cover all possible state transitions) [CWE-1245].

No security failure scenarios were written for HW class Insufficient Technical Documentation (CWE-1059) because it is too general to do so.

The HW class Reliance on Insufficiently Trustworthy Component (CWE-1357) has two security failure scenarios:

1. Attackers can compromise an SoC because it relies on the composition of IP blocks, one of which is untrustworthy [CWE-1357].
2. Attackers can compromise an SoC because it contains a vulnerable component that cannot be updated (e.g., firmware or ROM used in secure booting) [CWE-1329] [CWE-1277] [CWE-1310].

Under the non-HW class Violation of Secure Design Principles (CWE-657), there are three security failure scenarios:

1. An attacker can gain unauthorized access to IP blocks if the secure operation of an SoC is not achieved because the IP blocks are not securely and uniquely identified (e.g., missing, ignored, or insufficient identifiers) [CWE-1192].
2. A malicious agent can access sensitive assets because multiplexed resources (e.g., pins that are used by both trusted and untrusted agents but not at the same time) do not properly isolate accessible assets (e.g., between trusted and untrusted agents) [CWE-1189].
3. An attacker can use timing channels to infer sensitive data when a network-on-chip (NoC) does not provide proper isolation on the fabric and other resources between trusted and untrusted agents [CWE-1331].

One non-class HW CWE is a direct child of pillar Improper Adherence to Coding Standards (CWE-710). It has one security failure scenario:

1. An attacker can compromise a hardware state by writing to reserved bits (i.e., unused bits reserved for future functionality) that were covertly activated by developers for debugging or undocumented capabilities [CWE-1209].

4.3. Improper Check or Handling of Exceptional Conditions

The CWE Improper Adherence to Coding Standards (CWE-703) applies when a “product does not properly anticipate or handle exceptional conditions that rarely occur during normal operation of the product.”

The HW CWEs under this pillar occur within the following pillar/class hierarchy. The CWEs marked with * are HW CWEs.

CWE-703 P Improper Check or Handling of Exceptional Conditions

- CWE-1384 C Improper Handling of Physical or Environmental Conditions *

Figure 4 shows the digraph of hardware CWEs under this pillar with their parent-child relationships.

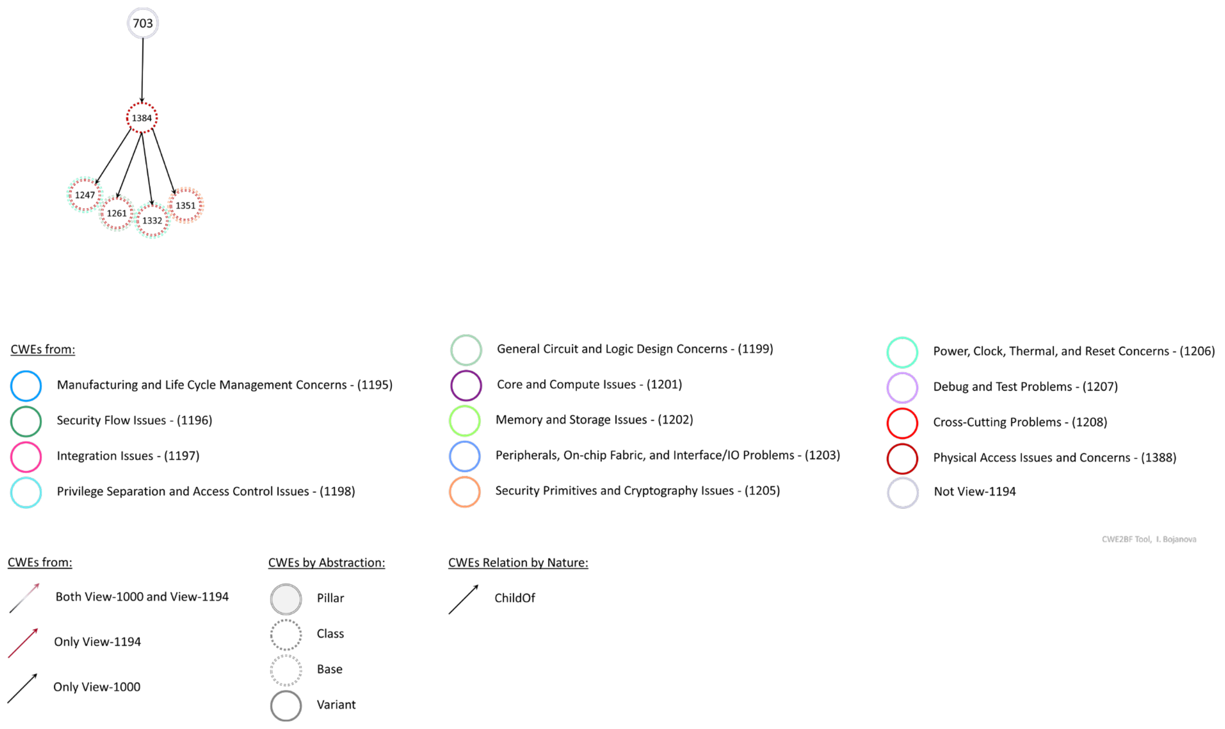

703
1384
1247
1261
1332
1351
CWEs from:
Manufacturing and Life Cycle Management Concerns - (1195)
Security Flow Issues - (1196)
Integration Issues - (1197)
Privilege Separation and Access Control Issues - (1198)
General Circuit and Logic Design Concerns - (1199)
Core and Compute Issues - (1201)
Memory and Storage Issues - (1202)
Peripherals, On-chip Fabric, and Interface/IO Problems - (1203)
Security Primitives and Cryptography Issues - (1205)
Power, Clock, Thermal, and Reset Concerns - (1206)
Debug and Test Problems - (1207)
Cross-Cutting Problems - (1208)
Physical Access Issues and Concerns - (1388)
Not View-1194
CWE2BF Tool, I. Bojanova
CWEs from:
Both View-1000 and View-1194
Only View-1194
Only View-1000
CWEs by Abstraction:
Pillar
Class
Base
Variant
CWEs Relation by Nature:
ChildOf

Fig. 4. HW CWE subgraph for pillar Improper Adherence to Coding Standards (CWE-703)

The HW class Improper Handling of Physical or Environmental Conditions (CWE-1384) has five security failure scenarios:

1. An attacker can leverage natural or maliciously created design-limit-exceeding physical or environmental conditions (e.g., atmospheric, electromagnetic interference, lasers, power variance, overclocking, component aging, cosmic radiation) to compromise the secure operations of a chip [CWE-1384].
    a. An attacker can compromise security functionality (e.g., secure boot) by introducing voltage and clock glitches (this can also happen naturally) [CWE-1247].
    b. An attacker can leverage the degradation of secure operations on a chip or a DoS due to single-event upsets (SEUs) (i.e., random bit flip errors) [CWE-1261].
    c. An attacker can bypass security-critical code by using fault injection techniques to skip security-critical instructions [CWE-1332].
    d. An attacker can cool hardware below the minimum design operating temperature to vary hardware behavior to compromise deployed security (e.g., power cycling not clearing volatile memory) [CWE-1351].

4.4. Improper Control of a Resource Through its Lifetime

The CWE Improper Control of a Resource Through its Lifetime (CWE-664) applies when a "product does not maintain or incorrectly maintains control over a resource throughout its lifetime of creation, use, and release."

The HW CWEs under this pillar occur within the following pillar/class hierarchy. No child class of the pillar is itself an HW CWE.

CWE-664 P Improper Control of a Resource Through its Lifetime

- CWE-400 C Uncontrolled Resource Consumption
- CWE-404 C Improper Resource Shutdown or Release
- CWE-610 C Externally Controlled Reference to a Resource in Another Sphere
- CWE-662 C Improper Synchronization
- CWE-665 C Improper Initialization
- CWE-668 C Exposure of Resource to Wrong Sphere
- CWE-669 C Incorrect Resource Transfer Between Spheres

Figure 5 shows the digraph of hardware CWEs under this pillar with their parent-child relationships.

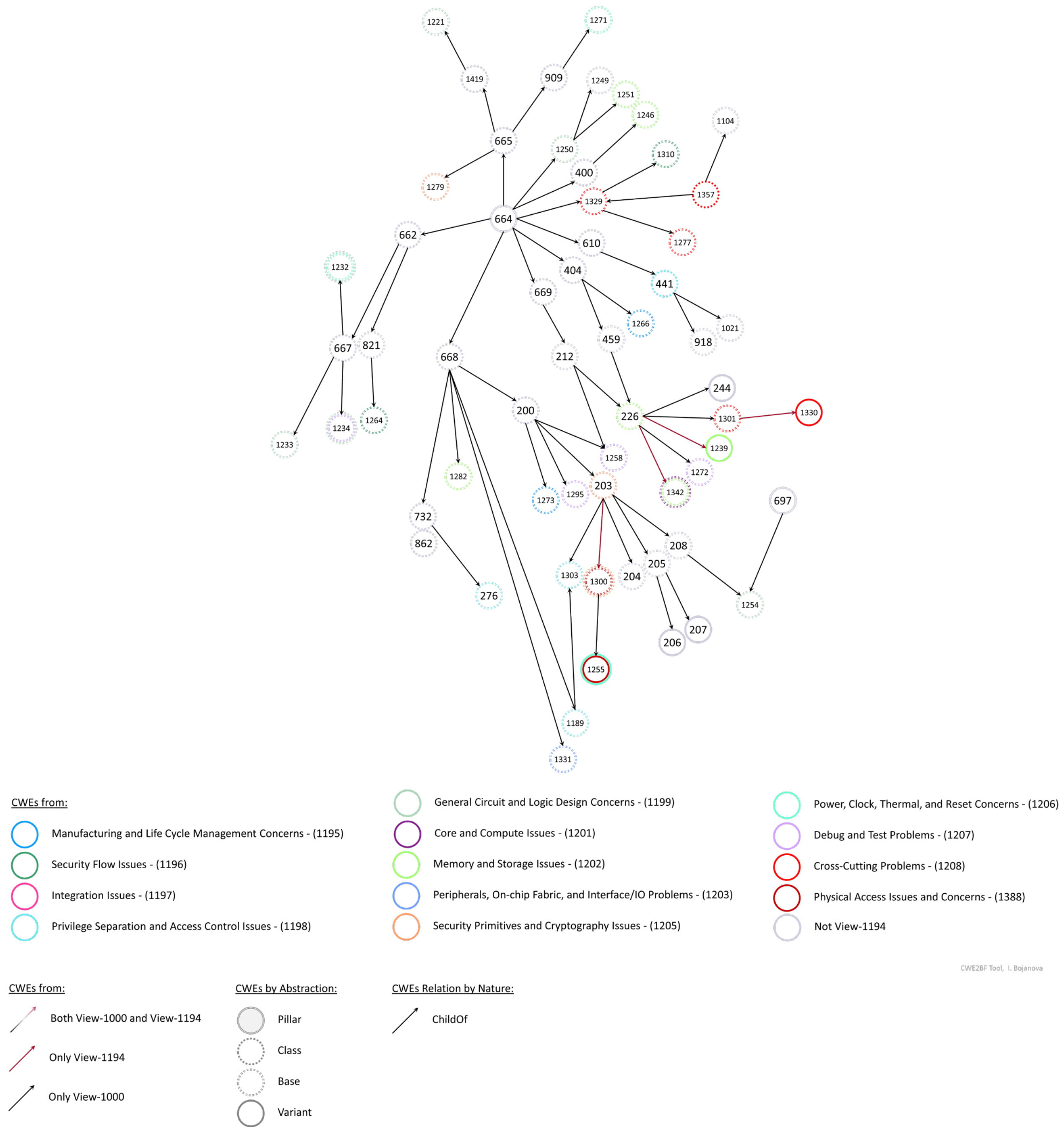

1221
1271
1419
909
1249
1251
1246
1104
665
1250
1310
400
1279
1329
1357
664
662
610
1277
1232
404
441
669
1266
1021
459
918
667
821
668
212
244
200
226
1330
1301
1264
1234
1233
1239
1258
1272
1282
203
1342
1273
1295
697
732
862
208
205
1303
1300
204
276
1254
207
206
1255
1189
1331
CWEs from:
Manufacturing and Life Cycle Management Concerns - (1195)
Security Flow Issues - (1196)
Integration Issues - (1197)
Privilege Separation and Access Control Issues - (1198)
General Circuit and Logic Design Concerns - (1199)
Core and Compute Issues - (1201)
Memory and Storage Issues - (1202)
Peripherals, On-chip Fabric, and Interface/IO Problems - (1203)
Security Primitives and Cryptography Issues - (1205)
Power, Clock, Thermal, and Reset Concerns - (1206)
Debug and Test Problems - (1207)
Cross-Cutting Problems - (1208)
Physical Access Issues and Concerns - (1388)
Not View-1194
CWE2BF Tool, I. Bojanova
CWEs from:
Both View-1000 and View-1194
Only View-1194
Only View-1000
CWEs by Abstraction:
Pillar
Class
Base
Variant
CWEs Relation by Nature:
ChildOf

Fig. 5. HW CWE subgraph for pillar Improper Control of a Resource Through its Lifetime (CWE-664)

Under the non-HW class Uncontrolled Resource Consumption (CWE-400), there is one security failure scenario:

1. An attacker can cause a premature failure of NVM by taking advantage of non-implemented or incorrectly implemented wear leveling operations (e.g., by repeated writing) [CWE-1246].

Under the non-HW class Improper Resource Shutdown or Release (CWE-404), there is one security failure scenario:

1. An attacker can retrieve sensitive information from decommissioned hardware that was not scrubbed of sensitive information [CWE-1266].

Under the non-HW class Externally Controlled Reference to a Resource in Another Sphere (CWE-610), there is one security failure scenario:

1. An attacker can violate access control by sending a message to a hardware component via an intermediary, whereby the message is interpreted by the recipient as having the privileges of the intermediary (not the original unprivileged sender) [CWE-441].

Under the non-HW class Improper Synchronization (CWE-662), there are four security failure scenarios.

1. An attacker can change system configuration information stored in lock-protected registers after a power state transition that causes improper lock behavior (e.g., making the lock programmable, clearing the lock, or resetting protected registers) [CWE-1232].
2. An attacker can violate access controls by directly changing system configurations protected by a register lock bit since the one-way lock that was properly set after system startup does not prevent the changes [CWE-1233].
3. An attacker can modify security-sensitive configuration information by using a debug mode to remove lock bit protections [CWE-1234].
4. An attacker can obtain access to sensitive data that is transmitted before security approval by taking advantage of errors in the separate control and data channels in hardware bus protocols [CWE-1264].

Under the non-HW class Improper Initialization (CWE-665), there are three security failure scenarios:

1. An attacker can read cryptographic output by taking advantage of weakened or broken cryptography that was encrypted before the cryptographic support units were ready (e.g., an external random number generator) [CWE-1279].
2. An attacker can compromise system security if register or IP parameter defaults (initialized at hardware reset) are incorrectly hard-coded with insecure values in the hardware description language code [CWE-1221].
3. An attacker can violate system security by taking advantage of an uninitialized security-critical register (e.g., before register initialization during system startup) [CWE-1271].

Under the non-HW class Exposure of Resource to Wrong Sphere (CWE-668), there are seven security failure scenarios:

1. An attacker can violate system security by changing security-sensitive and assumed-immutable data (e.g., golden digests) that are insecurely stored in writable memory instead of immutable memory (e.g., ROM, fuses, or one-time programmable memory [OTP]) [CWE-1282].

2. An attacker can unlock hardware (e.g., to enter debug mode) using leaked or stolen credentials that were often necessarily shared among multiple entities (e.g., for hardware products not created by a single company, via vertical integration) [CWE-1273].
3. An attacker can obtain sensitive information from debug messages that unnecessarily reveal security details, often reducing security by obscurity (e.g., location of password hashes) [CWE-1295].
4. An attacker can obtain security-relevant state information by observing different behaviors that are indicative of the hardware state (e.g., in timing, responses, and control flow) [CWE-203].
5. An attacker can obtain security-sensitive information by leveraging physical access to the hardware to measure phenomena (e.g., physical side channels, such as real-time power consumption) [CWE-1300] [CWE-1255].
6. An attacker can obtain sensitive data by evaluating and probing shared microarchitectural resources in contexts that should be isolated (e.g., caches and branch prediction logic) [CWE-1303].
7. Malicious software can take advantage of incorrectly assigned default permissions to obtain unauthorized access [CWE-276].

Under both the non-HW classes Exposure of Resource to Wrong Sphere (CWE-668) and Incorrect Resource Transfer Between Spheres (CWE-669), there is one HW CWE with one security failure scenario:

1. Attackers can obtain security-sensitive values from registers that are not cleared prior to entering debug mode [CWE-1258].

Under the non-HW class Incorrect Resource Transfer Between Spheres (CWE-669), there is one security failure scenario:

1. An attacker can infer sensitive data by observing discrepancies left behind by transient executions (i.e., speculative processing that was not needed and rolled back), detecting the transiency, and gaining evidence of the sensitive data values being processed [CWE-1420] [CWE-1421] [CWE-1422] [CWE-1423].

Under both the non-HW classes Improper Resource Shutdown or Release (CWE-404) and Incorrect Resource Transfer Between Spheres (CWE-669), there are four security failure scenarios:

1. Malicious software can read sensitive information from resources (e.g., registers) that were not cleared after use and that are made available due to a state change in the device (e.g., entering sleep or debug mode) or an execution change between privilege levels [CWE-226] [CWE-1272].
2. A malicious user of a hardware IP block can extract sensitive information stored in registers that were not zeroed after IP block use from a previous user (e.g., input/output registers) [CWE-1239].

3. An attacker can read sensitive data that was incompletely deleted or for which residual evidence or data remanence remains (e.g., performance optimizations that do not fully delete, physical properties that make data resistant to full deletion) [CWE-1301] [CWE-1330].

4. An attacker can take advantage of a process performing a transient execution (i.e., speculatively executed code) that leaves sensitive data in the microarchitectural state by provoking exceptions that allow the data to be read [CWE-1342].

There are HW CWEs that do not have an intervening class between them and pillar (CWE-664). They have one security failure scenario:

1. An attacker can violate system security by taking advantage of the need for multiple hardware components to keep local copies of a shared state (e.g., caches and MMUs) when they are unable to maintain full consistency [CWE-1250] [CWE-1251].

4.5. Incorrect Comparison

The CWE Incorrect Comparison (CWE-697) applies when a “product compares two entities in a security-relevant context, but the comparison is incorrect, which may lead to resultant weaknesses.” For example, the comparison:

- Checks one factor incorrectly
- Should consider multiple factors but does not check at least one of those factors at all
- Checks the wrong factor

The HW CWEs under this pillar occur within the CWE-697 P Incorrect Comparison pillar/class hierarchy.

Figure 6 shows the digraph of hardware CWEs under this pillar with their parent-child relationships.

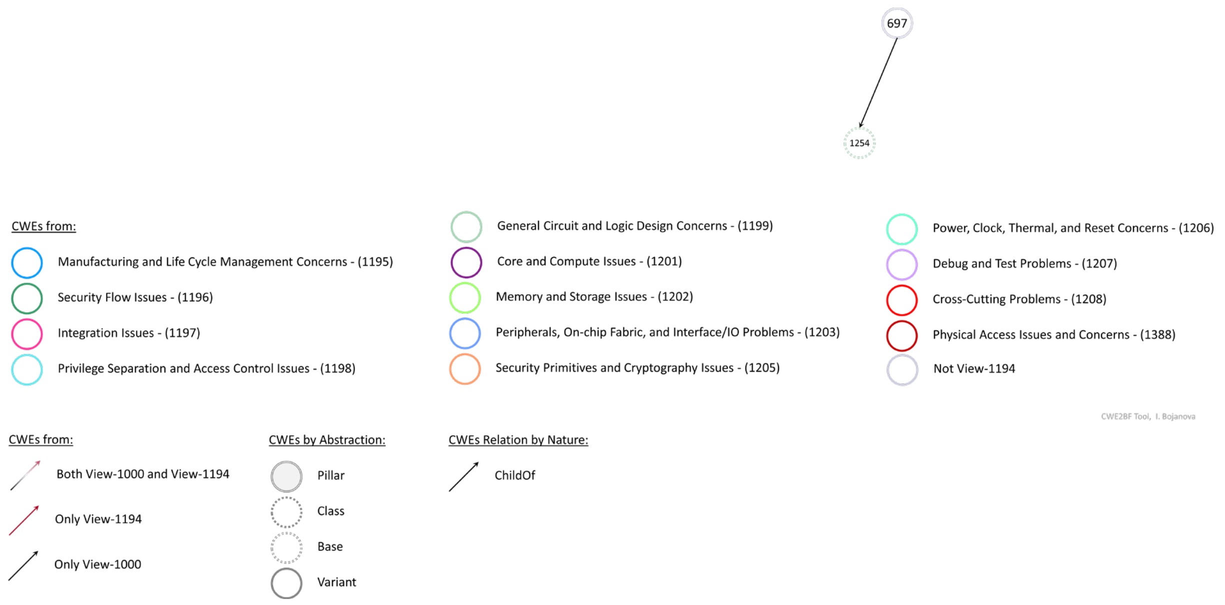

697
1254
CWEs from:
Manufacturing and Life Cycle Management Concerns - (1195)
Security Flow Issues - (1196)
Integration Issues - (1197)
Privilege Separation and Access Control Issues - (1198)
General Circuit and Logic Design Concerns - (1199)
Core and Compute Issues - (1201)
Memory and Storage Issues - (1202)
Peripherals, On-chip Fabric, and Interface/IO Problems - (1203)
Security Primitives and Cryptography Issues - (1205)
Power, Clock, Thermal, and Reset Concerns - (1206)
Debug and Test Problems - (1207)
Cross-Cutting Problems - (1208)
Physical Access Issues and Concerns - (1388)
Not View-1194
CWE2BF Tool, I. Bojanova
CWEs from:
Both View-1000 and View-1194
Only View-1194
Only View-1000
CWEs by Abstraction:
Pillar
Class
Base
Variant
CWEs Relation by Nature:
ChildOf

Fig. 6. HW CWE subgraph for pillar Incorrect Comparison (CWE-697)

The HW security failure scenario pertaining to this pillar is:

1. An attacker can make informed guesses of security credentials when evaluation of those credentials is performed iteratively as opposed to all at once (i.e., atomically) [CWE-1254].

4.6. Insufficient Control Flow Management

The CWE Insufficient Control Flow Management (CWE-691) applies when “the code does not sufficiently manage its control flow during execution, creating conditions in which the control flow can be modified in unexpected ways.”

The HW CWEs under this pillar occur within the following pillar/class hierarchy. No child class of the pillar is itself an HW CWE.

CWE-691 P Insufficient Control Flow Management

- CWE-362 C Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')
- CWE-662 C Improper Synchronization
    - CWE-667 C Improper Locking
- CWE-696 C Incorrect Behavior Order

Figure 7 shows the digraph of hardware CWEs under this pillar with their parent-child relationships.

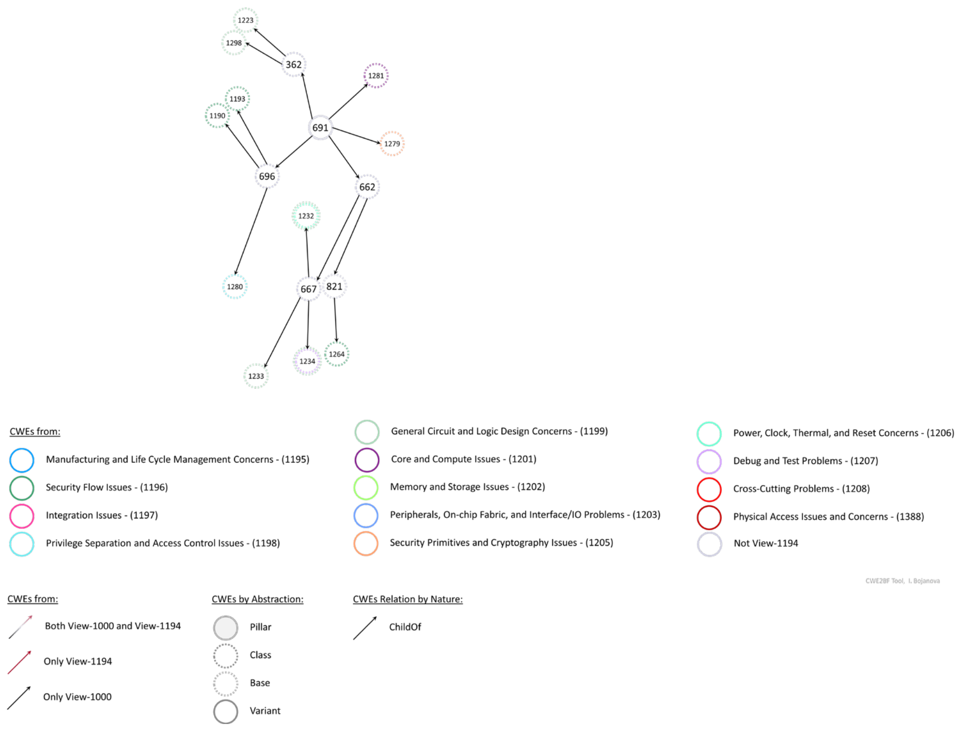

1223
1298
362
1281
1193
1190
691
1279
696
662
1232
1280
667
821
1264
1234
1233
CWEs from:
Manufacturing and Life Cycle Management Concerns - (1195)
Security Flow Issues - (1196)
Integration Issues - (1197)
Privilege Separation and Access Control Issues - (1198)
General Circuit and Logic Design Concerns - (1199)
Core and Compute Issues - (1201)
Memory and Storage Issues - (1202)
Peripherals, On-chip Fabric, and Interface/IO Problems - (1203)
Security Primitives and Cryptography Issues - (1205)
Power, Clock, Thermal, and Reset Concerns - (1206)
Debug and Test Problems - (1207)
Cross-Cutting Problems - (1208)
Physical Access Issues and Concerns - (1388)
Not View-1194
CWE2BF Tool, I. Bojanova
CWEs from:
Both View-1000 and View-1194
Only View-1194
Only View-1000
CWEs by Abstraction:
Pillar
Class
Base
Variant
CWEs Relation by Nature:
ChildOf

Fig. 7. HW CWE subgraph for pillar Insufficient Control Flow Management (CWE-691)

Under the non-HW class Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition') (CWE-362), there are two security failure scenarios:

1. Malicious software can violate the system security model by writing to write-once registers that typically hold system configuration data prior to trusted code writing to them [CWE-1223].
2. An attacker can circumvent security protections by taking advantage of a race condition in hardware logic [CWE-1298].

The non-HW class Improper Synchronization (CWE-662) has four security failure scenarios that were previously provided in Sec. 4.4. This is because CWE-662 also falls under pillar Improper Control of a Resource Through its Lifetime (CWE-664). The full graph in Fig. 1 shows the relationships.

Under the non-HW class Incorrect Behavior Order (CWE-696), there are three security failure scenarios:

1. An attacker can leverage an early boot IP with direct memory access (DMA) prior to security configuration settings being established in order to access security-sensitive data and potentially gain privileges by bypassing the operating system (OS) and bootloader [CWE-1190].
2. An attacker can leverage an untrusted IP or peripheral microcontroller after system reset to access memory and fabric (e.g., to obtain privileges or read sensitive data) prior to trusted firmware asserting security controls during the boot sequence [CWE-1193].

3. A malicious agent can gain access to a protected asset if the hardware-based access control check does not complete prior to the asset being accessed [CWE-1280].

The HW child of pillar CWE-691 has one security failure scenario:

1. Malicious code can cause undesirable processor behavior (e.g., lock a processor) by executing a special sequence of instructions [CWE-1281].

4.7. Protection Mechanism Failure

The CWE Protection Mechanism Failure (CWE-693) applies when:

> The product does not use or incorrectly uses a protection mechanism that provides sufficient defense against directed attacks against the product. This weakness covers three distinct situations. A 'missing' protection mechanism occurs when the application does not define any mechanism against a certain class of attack. An 'insufficient' protection mechanism might provide some defenses - for example, against the most common attacks - but it does not protect against everything that is intended. Finally, an 'ignored' mechanism occurs when a mechanism is available and in active use within the product, but the developer has not applied it in some code path.

There are 15 HW CWEs under this pillar. They occur within the following pillar/class hierarchy. No child class of the pillar is itself an HW CWEs.

CWE-693 P Protection Mechanism Failure

- CWE-311 C Missing Encryption of Sensitive Data
- CWE-327 C Use of a Broken or Risky Cryptographic Algorithm
- CWE-330 C Use of Insufficiently Random Value

Figure 8 shows the digraph of hardware CWEs under this pillar with their parent-child relationships.

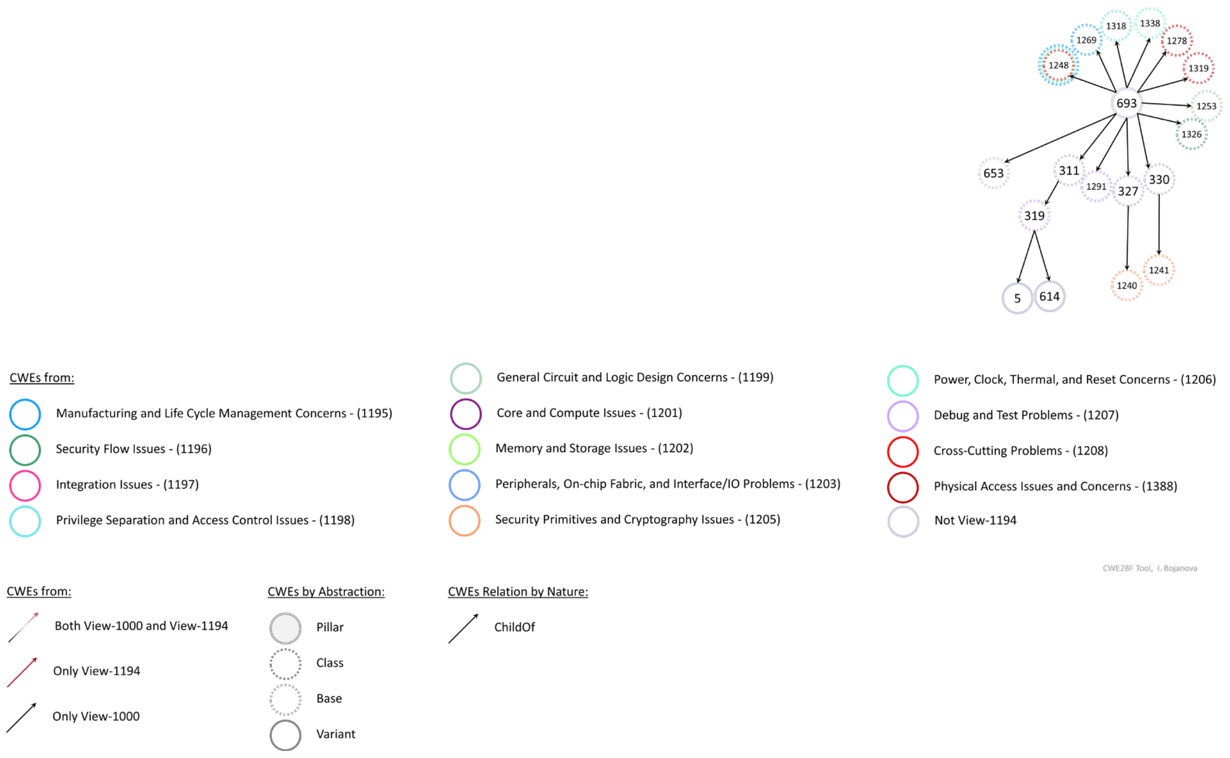

693
1248
1269
1318
1338
1278
1319
1253
1326
653
311
1291
327
330
319
5
614
1240
1241
CWEs from:
Manufacturing and Life Cycle Management Concerns - (1195)
Security Flow Issues - (1196)
Integration Issues - (1197)
Privilege Separation and Access Control Issues - (1198)
General Circuit and Logic Design Concerns - (1199)
Core and Compute Issues - (1201)
Memory and Storage Issues - (1202)
Peripherals, On-chip Fabric, and Interface/IO Problems - (1203)
Security Primitives and Cryptography Issues - (1205)
Power, Clock, Thermal, and Reset Concerns - (1206)
Debug and Test Problems - (1207)
Cross-Cutting Problems - (1208)
Physical Access Issues and Concerns - (1388)
Not View-1194
CWE2BF Tool, I. Bojanova
CWEs from:
Both View-1000 and View-1194
Only View-1194
Only View-1000
CWEs by Abstraction:
Pillar
Class
Base
Variant
CWEs Relation by Nature:
ChildOf

Fig. 8. HW CWE subgraph for pillar Protection Mechanism Failure (CWE-693)

Under the non-HW class Missing Encryption of Sensitive Data (CWE-311), there is one security failure scenario:

1. An attacker may gain access to sensitive information if it is transmitted unencrypted through on-chip component interconnects or external debug channels (e.g., JTAG debug port) [CWE-319].

Under the non-HW class Use of a Broken or Risky Cryptographic Algorithm (CWE-327), there is one security failure scenario:

1. An attacker can read encrypted information since HW-implemented cryptographic primitives may not be easily patchable or upgradeable, resulting in a weakening of cryptographic services over time as the computational power of attackers increases and vulnerabilities are discovered that weaken implemented algorithms [CWE-1240].

Under the non-HW class Use of Insufficiently Random Values (CWE-330), there is one security failure scenario:

1. An attacker may break encryption by leveraging the ability to predict generated ‘random’ numbers that come from pseudorandom number generators (RNGs) as opposed to hardware-based true random number generators (TRNGs) [CWE-1241].

The non-class children of pillar CWE-693 have the following nine security failure scenarios:

1. An attack can leverage a security-sensitive hardware module that may fail due to semiconductor defects that already existed in a new chip or that occurred over time

(e.g. due to thermal/electrical stress). Such failures can freeze signals to either 0 or 1. [CWE-1248].

2. An attacker can blow a fuse to put a chip into an insecure state with one-directional fuses on chips used to permanently set a configuration (e.g., 'Manufacturing Complete') when such fuses incorrectly implement a reverse security logic [CWE-1253].
3. An attacker can gain unauthorized capabilities (e.g., bypass cryptographic checks, read and change an internal state, and adjust system configurations) when a chip is not set to a production configuration, thereby allowing debug capabilities [CWE-1269].
4. An attacker can read confidential information from a chip (e.g., secret keys, device identifiers, proprietary code, and circuit designs) with imaging technology (e.g., x-ray microscopy and scanning electron microscopes) after the removal of chip packaging and individual integrated circuit layers [CWE-1278].
5. An attacker can run leaked debug firmware on a chip and gain greater insight into the inner workings and state of the chip if both the debug and production firmware are signed with the same public key [CWE-1291].
6. An attacker can bypass security by leveraging peripherals and chip components that require the transfer of information for security features (e.g., privileges and immutable identity) but that are connected to on-chip fabrics or buses that do not support those features [CWE-1318].
7. An attacker can generate magnetic pulses to induce temporary faults on a chip (known as electromagnetic fault injection), thereby circumventing or changing security functionality (e.g., bypassing security features, reading confidential information, changing program flow, or perturbing RNGs) [CWE-1319].
8. An attacker can compromise secure boot capabilities and execute their choice of code by modifying memory or fuses that should have been made immutable [CWE-1326].
9. Malicious software can execute code to trigger overheating on chips that contain inadequate thermal protection (e.g., heat sensors and cooling capabilities), resulting in temporary DoS, permanent failure ("bricking"), reliability issues, and physical safety hazards [CWE-1338].

5. Categories of Hardware Design Weaknesses

The HW CWE SIG groups HW weaknesses into 13 categories that describe where a security problem may exist in an HW design. This section presents these categories and the associated HW CWEs.

5.1. Core and Compute Issues

Weaknesses in the category Core and Compute Issues (CWE-1201) are "typically associated with CPUs, Graphics, Vision, AI, FPGA, and microcontrollers." There are three HW CWEs in this category, none of which are classes.

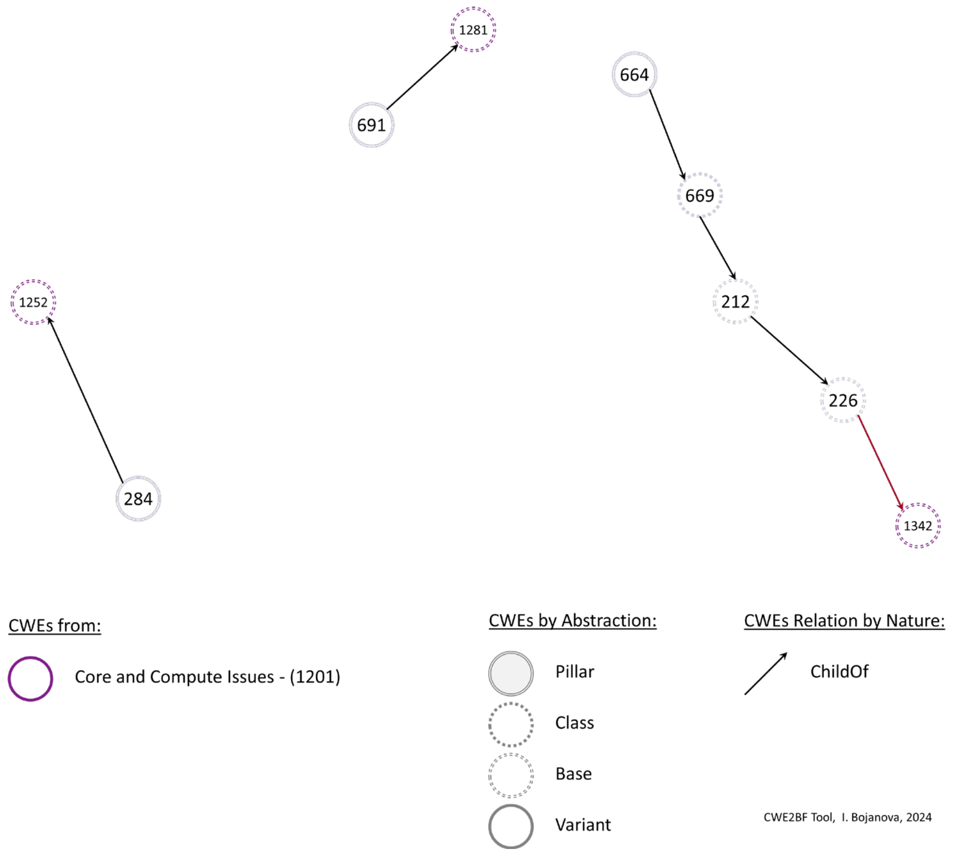
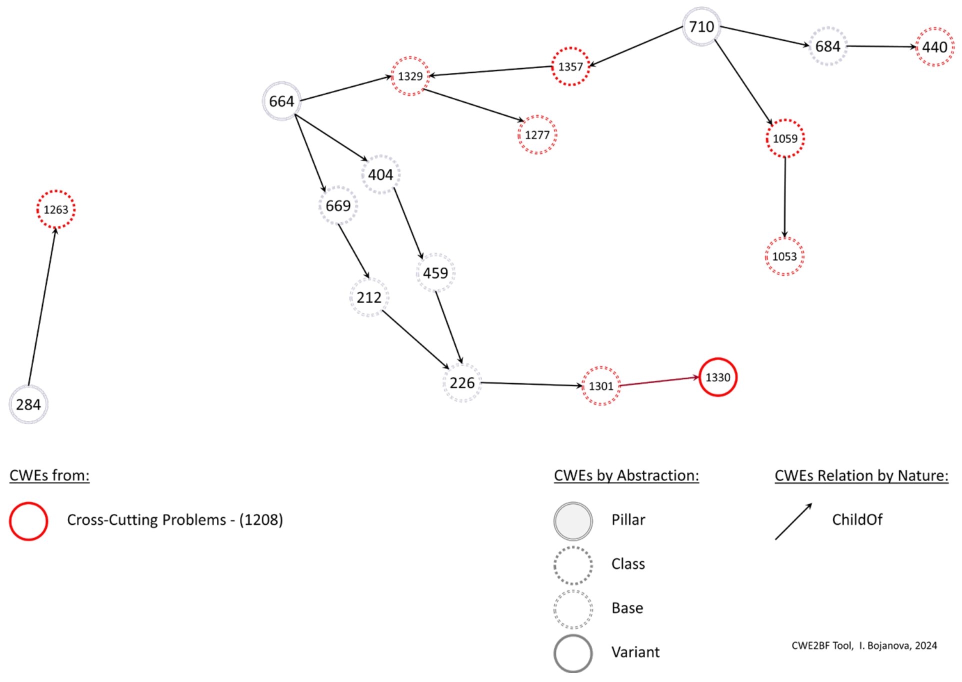


Fig. 9. HW CWEs under the category Core and Compute Issues (CWE-1201)

5.2. Cross-Cutting Problems

Weaknesses in the category Cross-Cutting Problems (CWE-1208) can "arise in multiple areas of hardware design or apply to a wide cross-section of components." There are nine HW CWEs in this category. Three are classes Insufficient Technical Documentation (CWE-1059), Improper

848 Physical Access Control (CWE-1263), and Reliance on Insufficiently Trustworthy Component
849 (CWE-1357).

850


851 **Fig. 10. HW CWEs under the category Cross-Cutting Problems (CWE-1208)**

5.3. Debug and Test Problems

Weaknesses in the category Debug and Test Problems (CWE-1207) are "related to hardware debug and test interfaces such as JTAG and scan chain)." There are 12 HW CWEs in this category, none of which are classes.

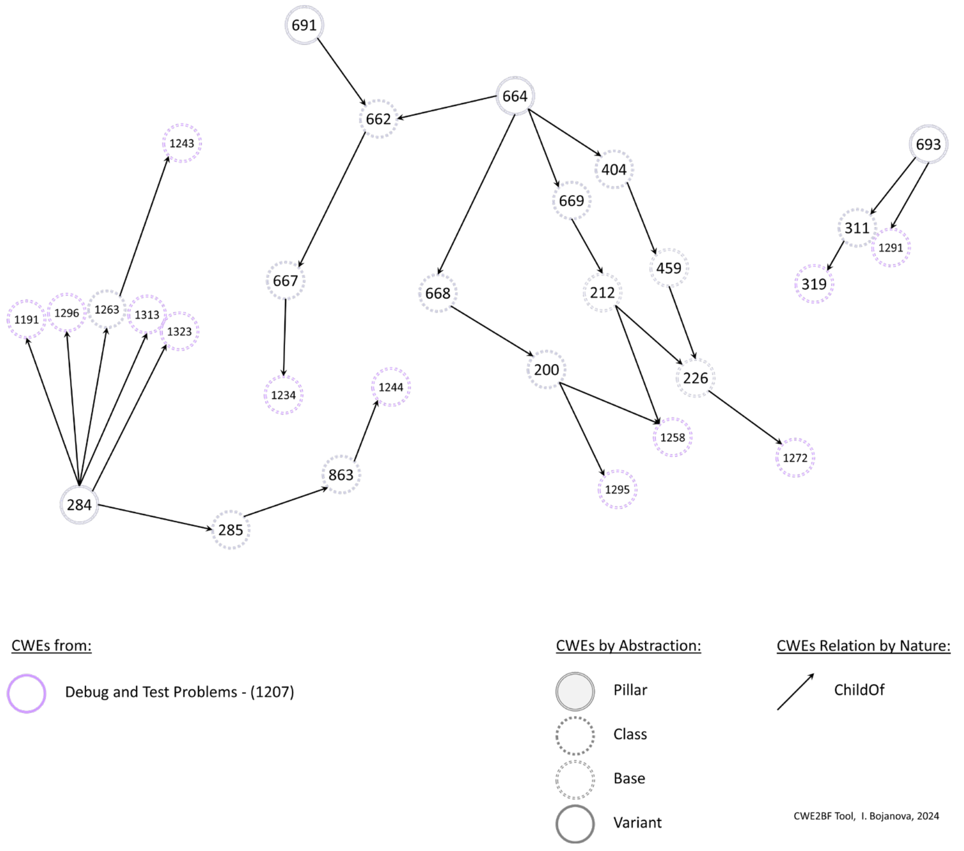


Fig. 11. HW CWEs under the category Debug and Test Problems (CWE-1207)

5.4. General Circuit and Logic Design Concerns

Weaknesses in the category General Circuit and Logic Design Concerns (CWE-1199) are "related to hardware-circuit design and logic (e.g., CMOS transistors, finite state machines, and registers) as well as issues related to hardware description languages such as System Verilog and VHDL).” There are 14 HW CWEs in this category, none of which are classes.

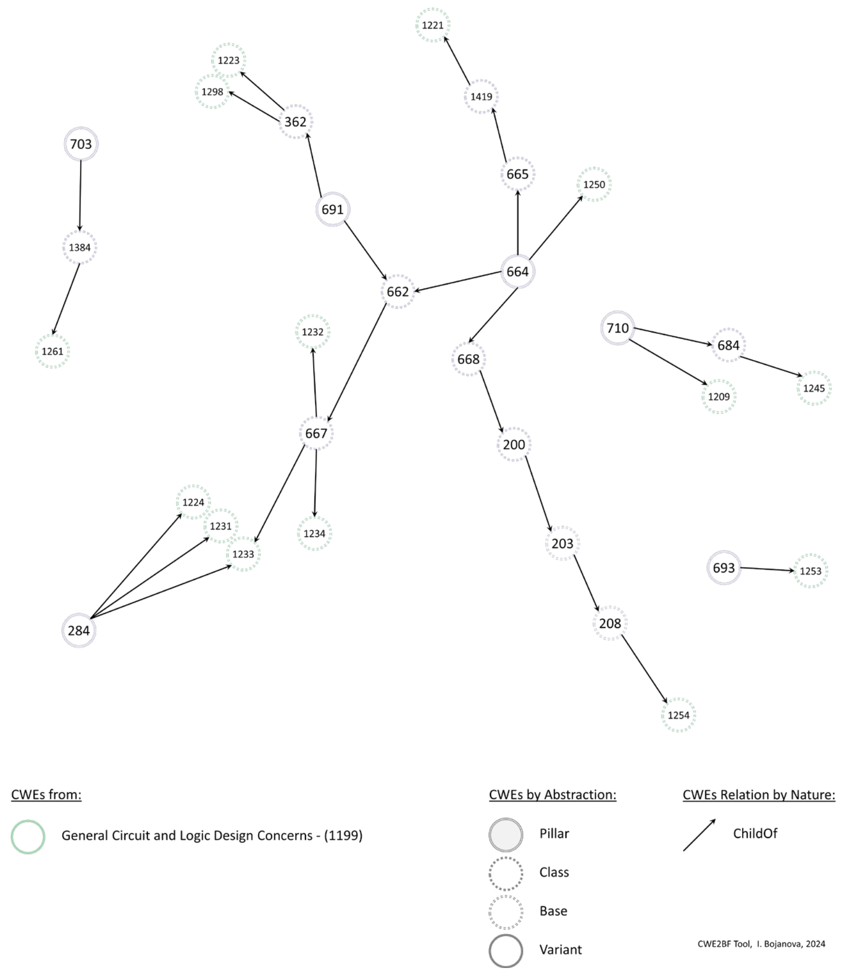


Fig. 12. HW CWEs under the category General Circuit and Logic Design Concerns (CWE-1199)

5.5. Integration Issues

Weaknesses in the category Integration Issues (CWE-1197) arise from the integration of multiple hardware IP cores, SoC subsystem interactions, or hardware platform subsystem interactions. There is only one HW CWE in this category.

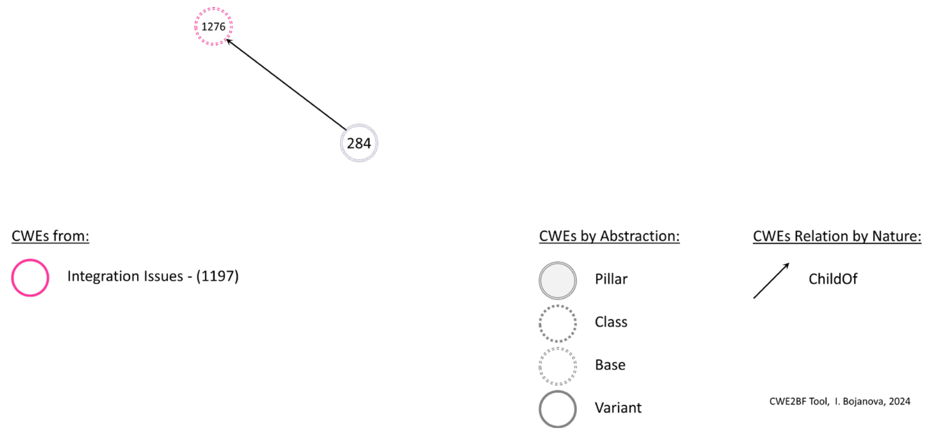


Fig. 13. HW CWEs under the category Integration Issues (CWE-1197)

5.6. Manufacturing and Life Cycle Management Concerns

Weaknesses in the category Manufacturing and Life Cycle Management Concerns (CWE-1195) are “root-caused to defects that arise in the semiconductor-manufacturing process or during the life cycle and supply chain." There are six HW CWEs in this category, one of which is class Insufficient Technical Documentation (CWE-1059).

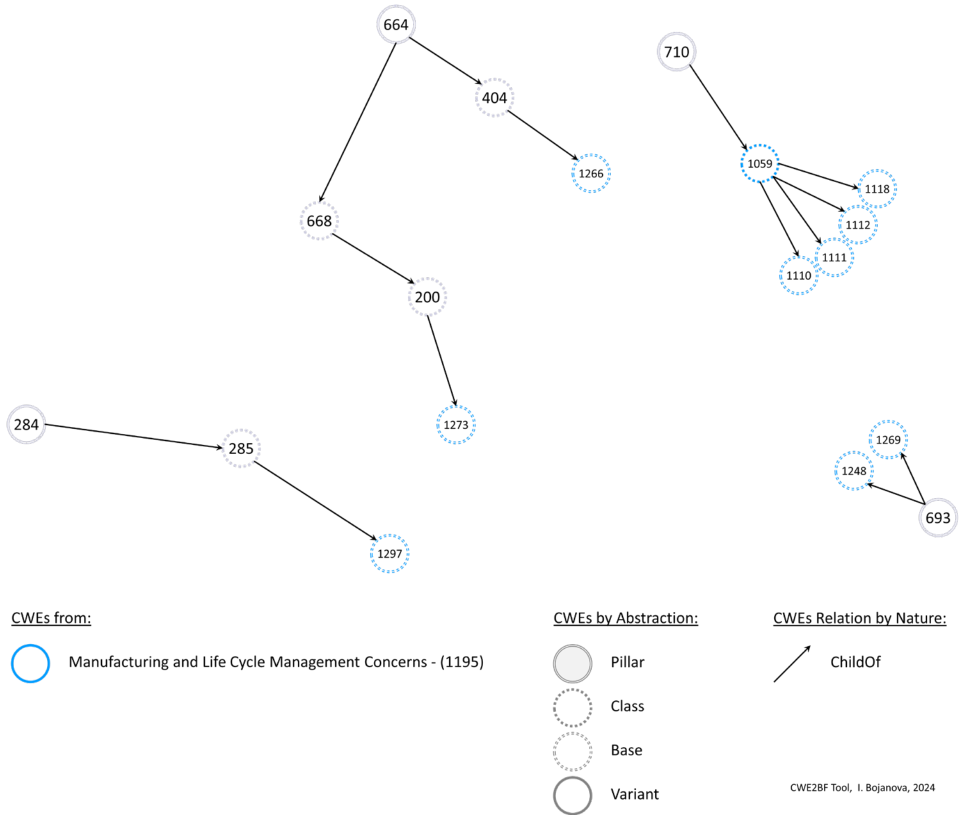


Fig. 14. HW CWEs under the category Manufacturing and Life Cycle Management Concerns (CWE-1195)

5.7. Memory and Storage Issues

Weaknesses in the category Memory and Storage Issues (CWE-1202) are “typically associated with memory (e.g., DRAM, SRAM) and storage technologies (e.g., NAND Flash, OTP, EEPROM, and eMMC).” There are seven HW CWEs in this category, none of which are classes.

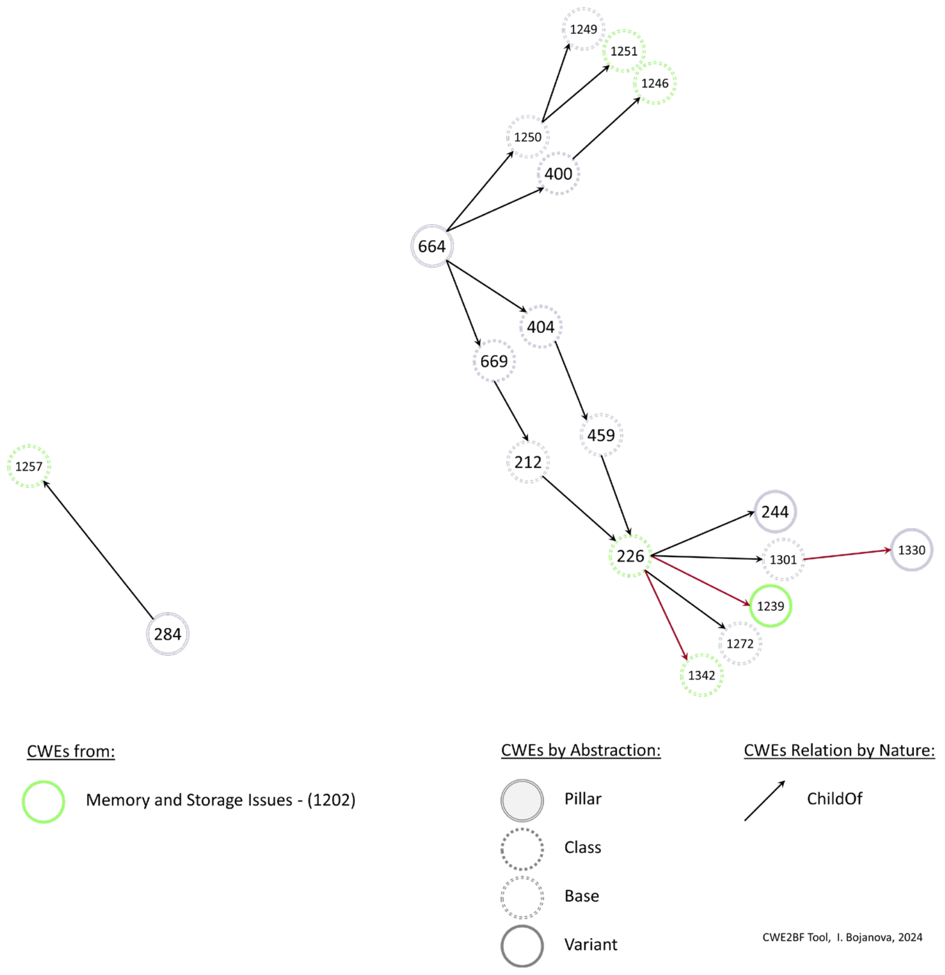


Fig. 15. HW CWEs under the category Memory and Storage Issues (CWE-1202)

5.8. Peripherals, On-chip Fabric, and Interface/IO Problems

Weaknesses in the category Peripherals, On-chip Fabric, and Interface/IO Problems (CWE-1203) are “related to hardware security problems that apply to peripheral devices, IO interfaces, on-chip interconnects, NoC, and buses. For example, this category includes issues related to design of hardware interconnect and/or protocols, such as PCIe, USB, SMBUS, general-purpose IO pins, and user-input peripherals such as mouse and keyboard.” There are six HW CWEs in this category, none of which are classes.

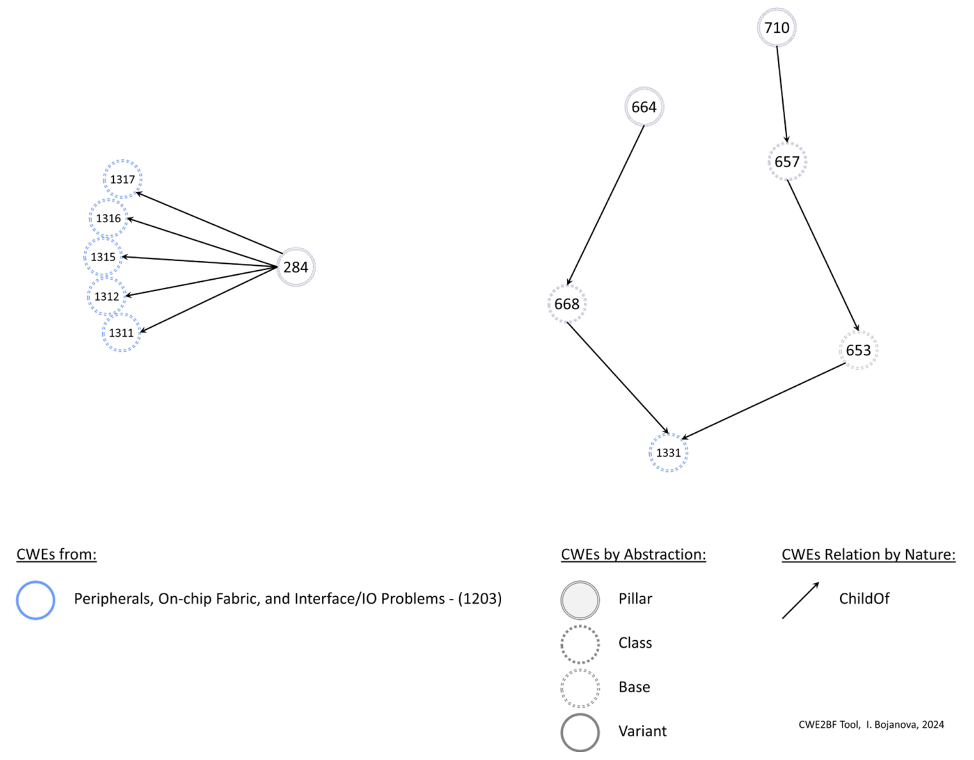


Fig. 16. HW CWEs under the category Peripherals, On-chip Fabric, and Interface/IO Problems (CWE-1203)

5.9. Physical Access Issues and Concerns

Weaknesses in the category Physical Access Issues and Concerns (CWE-1388) are related to physical access concerns. There are 10 HW CWEs in this category, one of which is class Improper Handling of Physical or Environmental Conditions (CWE-1384).

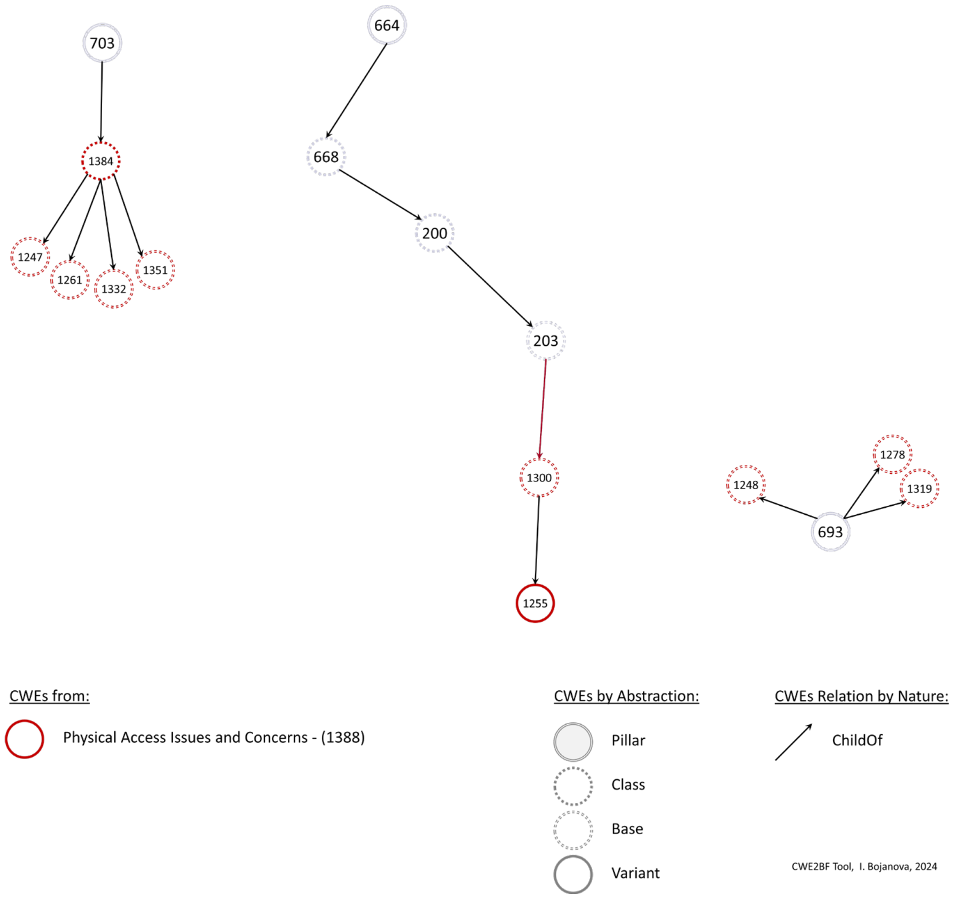


Fig. 17. Figure 18. HW CWEs under the category Physical Access Issues and Concerns (CWE-1388)

5.10. Power, Clock, Thermal, and Reset Concerns

Weaknesses in the category Power, Clock, Thermal, and Reset Concerns (CWE-1206) are “related to system power, voltage, current, temperature, clocks, system state saving/restoring, and resets at the platform and SoC level." There are 11 HW CWEs in this category, none of which are classes.

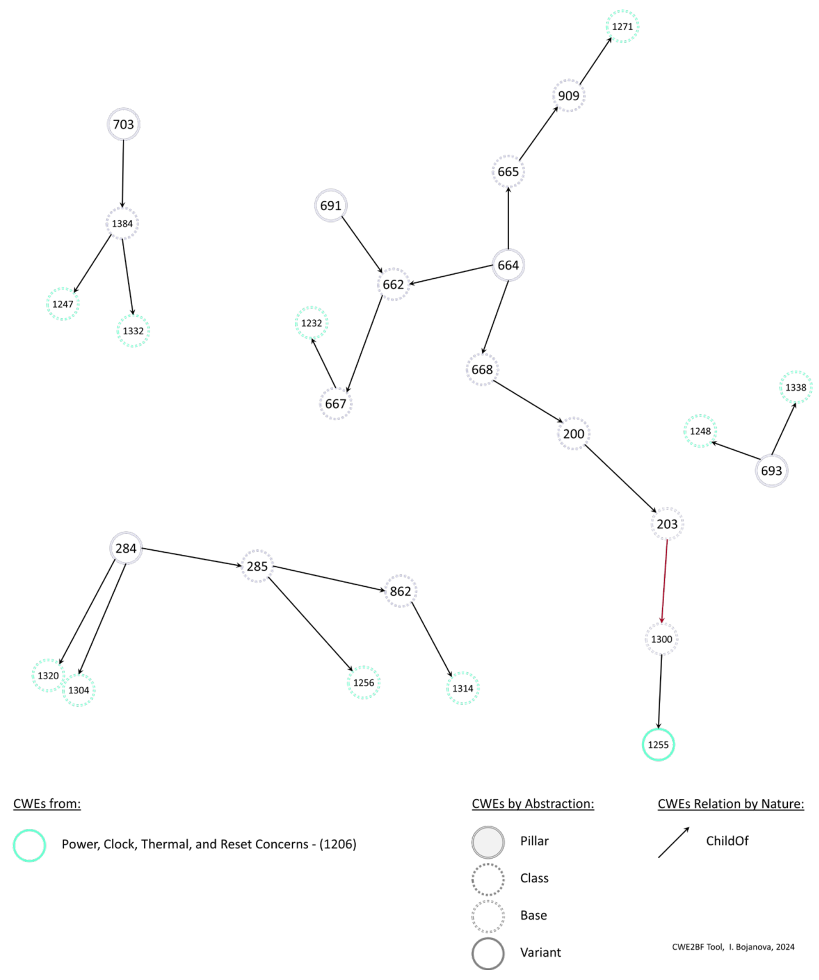


Fig. 18. HW CWEs under the category Power, Clock, Thermal, and Reset Concerns (CWE-1206)

5.11. Privilege Separation and Access Control Issues

Weaknesses in the category Privilege Separation and Access Control Issues (CWE-1198) are "related to features and mechanisms providing hardware-based isolation and access control (e.g., identity, policy, locking control) of sensitive shared hardware resources, such as registers and fuses." There are 23 HW CWEs in this category, two of which are classes Unintended Proxy or Intermediary ('Confused Deputy') (CWE-441) and Insecure Security Identifier Mechanism (CWE-1294).

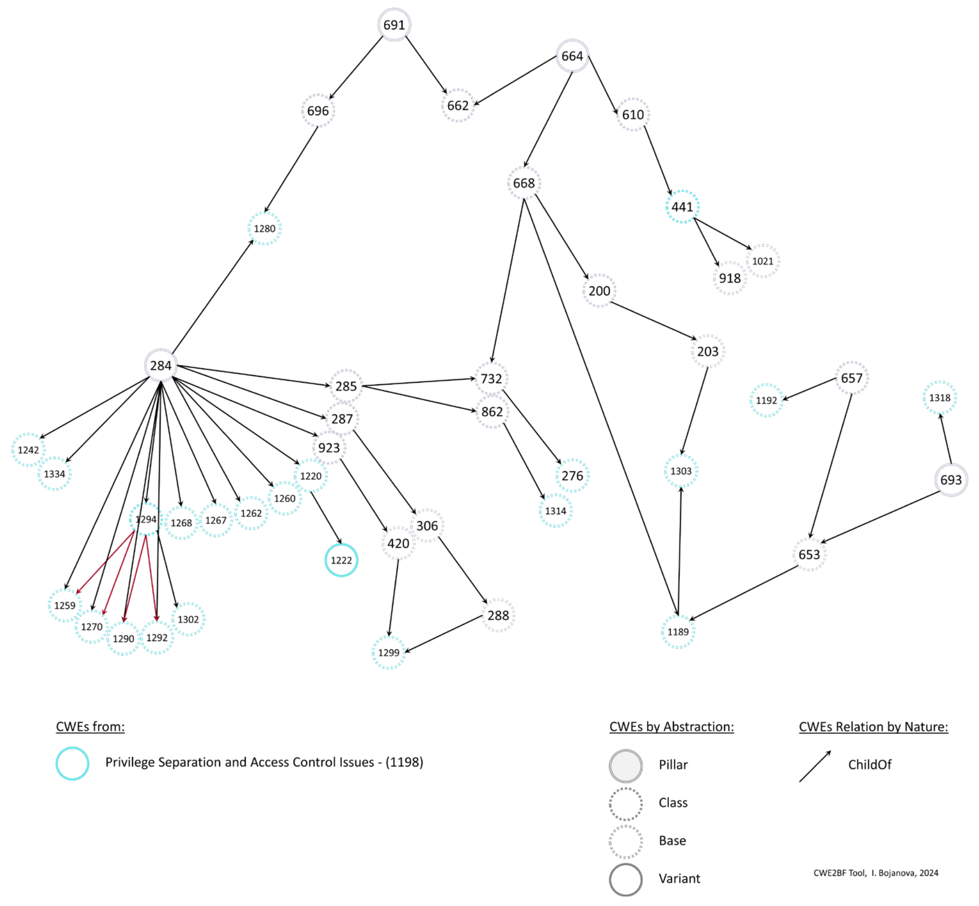


Fig. 19. HW CWEs under the category Privilege Separation and Access Control Issues (CWE-1198)

5.12. Security Flow Issues

Weaknesses in the category Security Flow Issues (CWE-1196) are "related to improper design of full-system security flows, including but not limited to secure boot, secure update, and hardware-device attestation." There are eight HW CWEs in this category, none of which are classes.

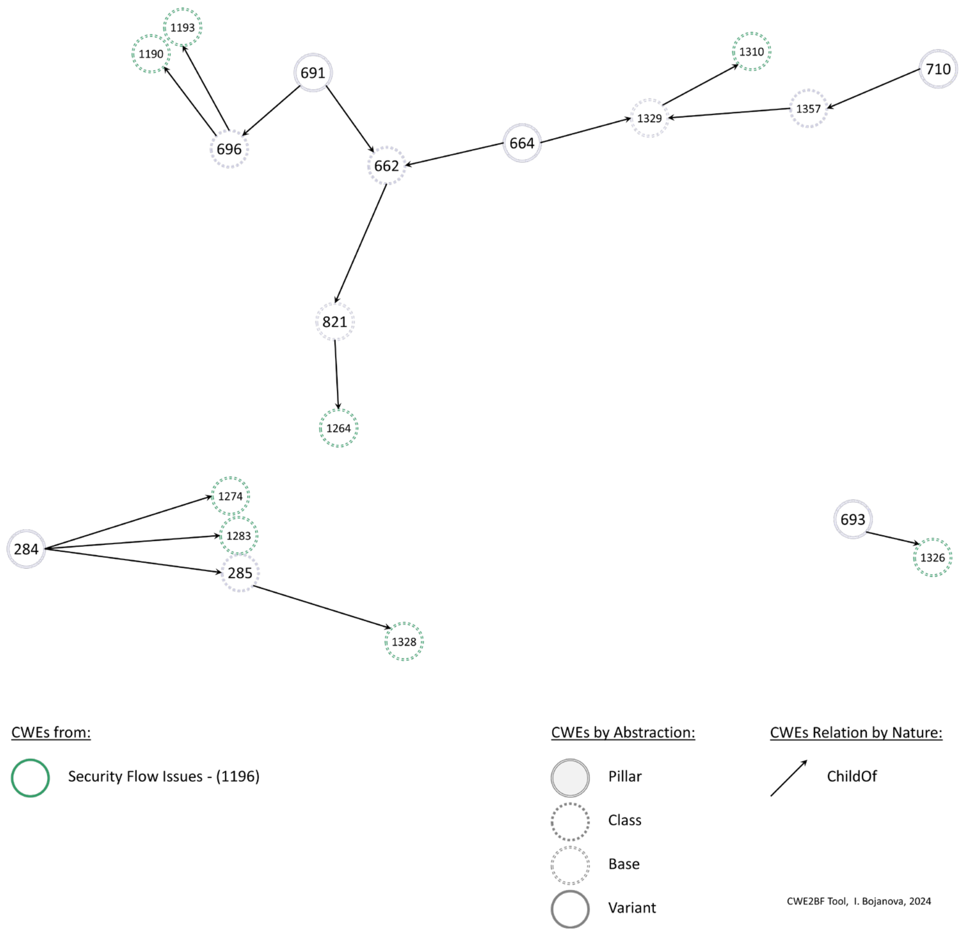


Fig. 20. HW CWEs under the category Security Flow Issues (CWE-1196)

5.13. Security Primitives and Cryptography Issues

Weaknesses in the category Security Primitives and Cryptography Issues (CWE-1205) are "related to hardware implementations of cryptographic protocols and other hardware-security primitives, such as physical unclonable functions (PUFs) and random number generators (RNGs).” There are seven HW CWEs in this category, none of which are classes.

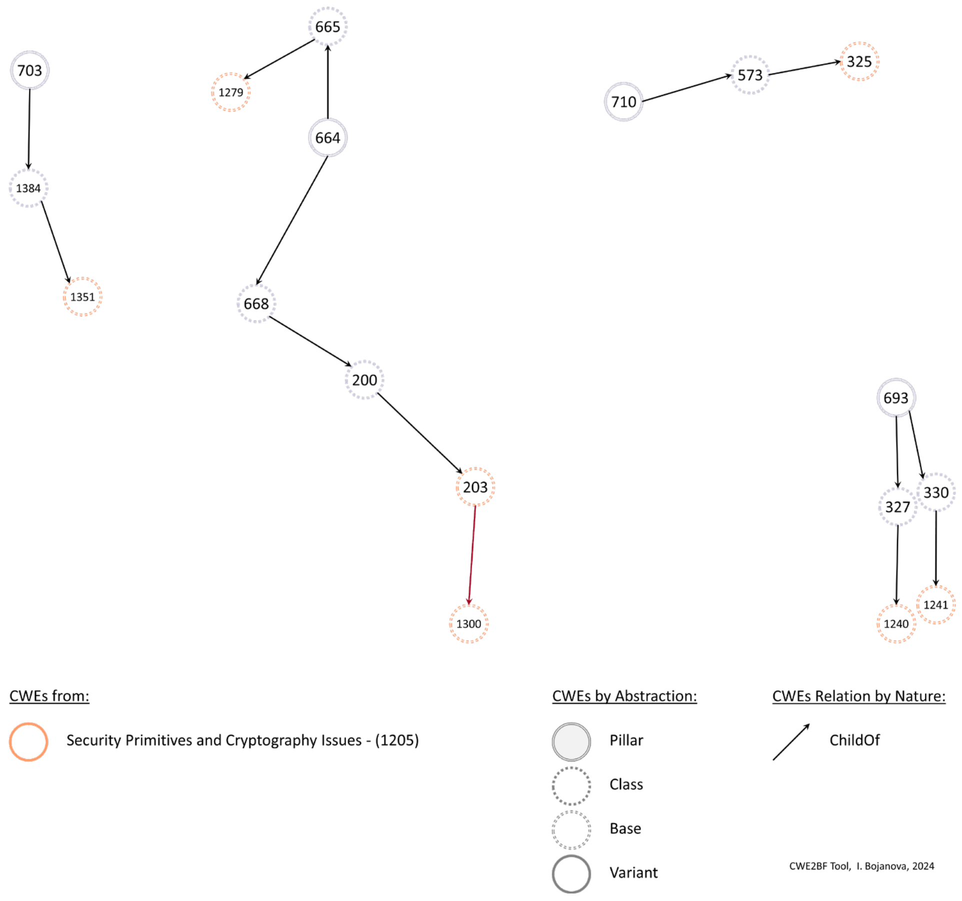


Fig. 21. HW CWEs under the category Security Primitives and Cryptography Issues (CWE-1205)

6. Comparison With Software Weaknesses

As presented in Sec. 2.4.3, the Weaknesses for Simplified Mapping of Published Vulnerabilities view (CWE-1003) includes the CWEs that cover the majority of CVEs. As presented in Sec. 2.4.1, the Hardware Design view (CWE-1194) contains the HW CWEs.

There are only three CWEs that overlap in View-1003 and View-1194: CWE-203, CWE-276, and CWE-319. The have the following View-1194 categories:

1. Observable Discrepancy (CWE-203) is in View-1194 category Security Primitives and Cryptography Issues (CWE-1205).
2. Incorrect Default Permissions (CWE-276) is in View-1194 category Privilege Separation and Access Control Issues (CWE-1198).
3. Cleartext Transmission of Sensitive Information (CWE-319) is in View-1194 category Debug and Test Problems (CWE-1207).

Figure 22 shows the complete HW CWE graph created using View-1000 and View-1194 (from Fig. 1) with the View-1003 software CWEs added and highlighted in dark purple. Twenty of these CWEs occur within the HW CWE graph even though 17 of them are not HW CWEs. These 17 are intermediary CWEs that connect an HW CWE with its respective pillars.

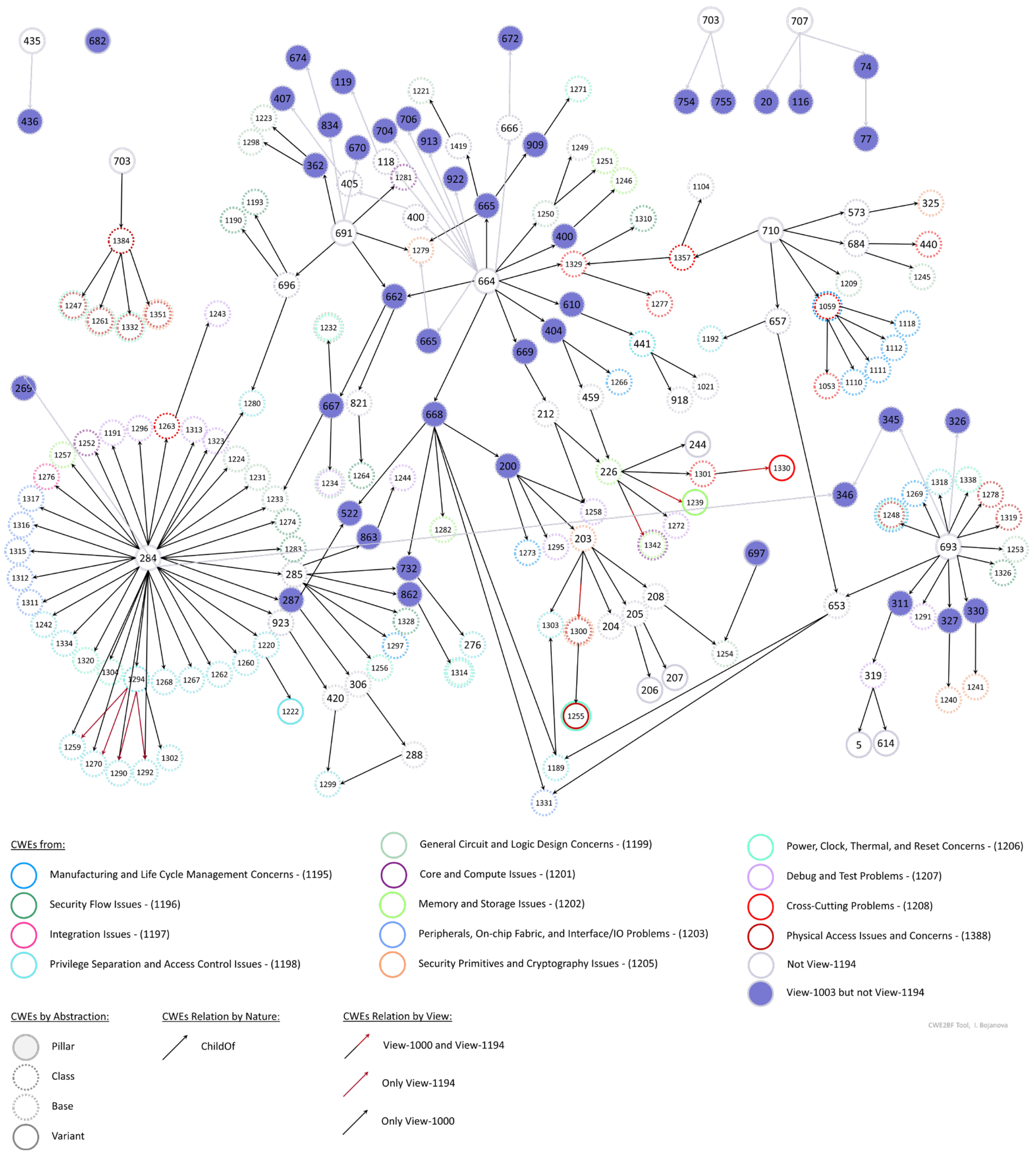

CWEs from:
Manufacturing and Life Cycle Management Concerns - (1195)
Security Flow Issues - (1196)
Integration Issues - (1197)
Privilege Separation and Access Control Issues - (1198)
General Circuit and Logic Design Concerns - (1199)
Core and Compute Issues - (1201)
Memory and Storage Issues - (1202)
Peripherals, On-chip Fabric, and Interface/IO Problems - (1203)
Security Primitives and Cryptography Issues - (1205)
Power, Clock, Thermal, and Reset Concerns - (1206)
Debug and Test Problems - (1207)
Cross-Cutting Problems - (1208)
Physical Access Issues and Concerns - (1388)
Not View-1194
View-1003 but not View-1194
CWEs by Abstraction:
Pillar
Class
Base
Variant
CWEs Relation by Nature:
ChildOf
CWEs Relation by View:
View-1000 and View-1194
Only View-1194
Only View-1000
CWE2BF Tool, I. Bojanova

Fig. 22. HW CWE complete graph with View-1003 pillar and class CWEs that are not in View-1194 highlighted

Figure 23 shows the complete HW CWE graph created using View-1000 and View-1194 (from Fig. 1) with the three CWEs that occur both in View-1003 and View-1194 highlighted in purple.

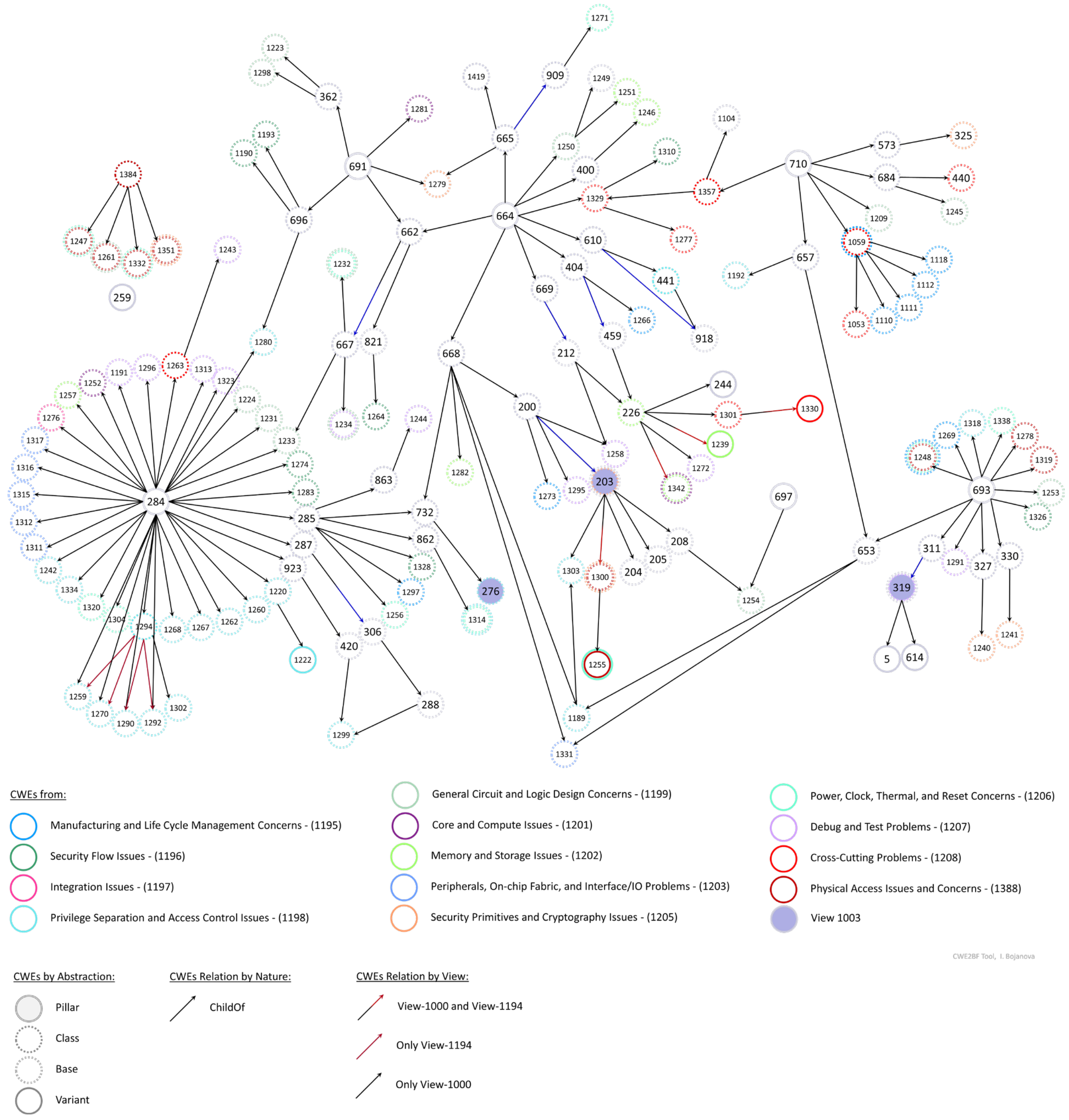

CWEs from:
Manufacturing and Life Cycle Management Concerns - (1195)
Security Flow Issues - (1196)
Integration Issues - (1197)
Privilege Separation and Access Control Issues - (1198)
General Circuit and Logic Design Concerns - (1199)
Core and Compute Issues - (1201)
Memory and Storage Issues - (1202)
Peripherals, On-chip Fabric, and Interface/IO Problems - (1203)
Security Primitives and Cryptography Issues - (1205)
Power, Clock, Thermal, and Reset Concerns - (1206)
Debug and Test Problems - (1207)
Cross-Cutting Problems - (1208)
Physical Access Issues and Concerns - (1388)
View 1003
CWE2BF Tool, I. Bojanova
CWEs by Abstraction:
Pillar
Class
Base
Variant
CWEs Relation by Nature:
ChildOf
CWEs Relation by View:
View-1000 and View-1194
Only View-1194
Only View-1000

Fig. 23. HW CWE complete graph with View-1003 base CWEs that overlap with View-1194 highlighted

Figure 24 shows the complete HW CWE graph with memory-related weaknesses darkly shaded in purple. These may be candidates to be analyzed for addition as HW CWEs if firmware (including microcode) weaknesses are considered HW weaknesses.

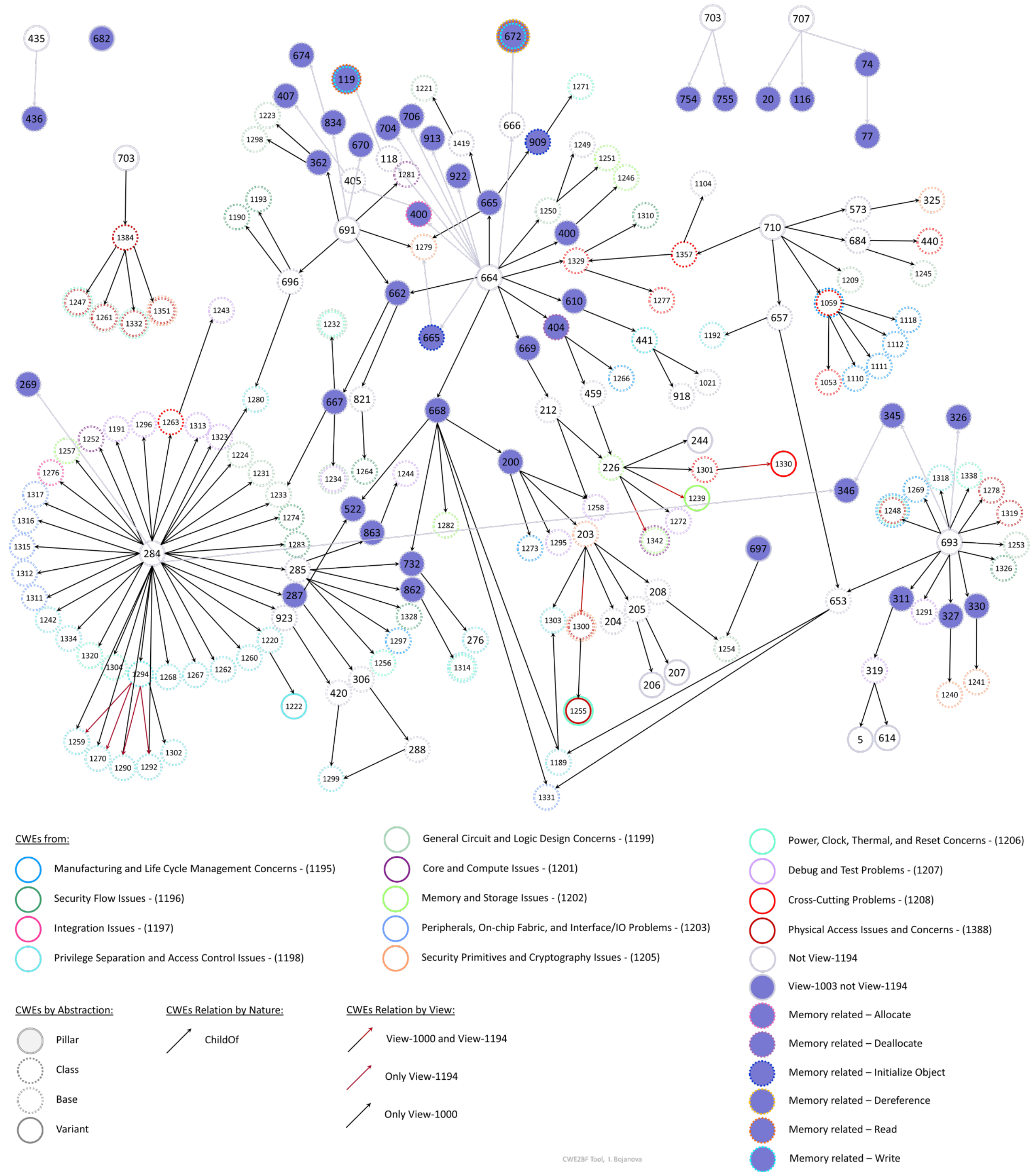

CWEs from:
Manufacturing and Life Cycle Management Concerns - (1195)
Security Flow Issues - (1196)
Integration Issues - (1197)
Privilege Separation and Access Control Issues - (1198)
General Circuit and Logic Design Concerns - (1199)
Core and Compute Issues - (1201)
Memory and Storage Issues - (1202)
Peripherals, On-chip Fabric, and Interface/IO Problems - (1203)
Security Primitives and Cryptography Issues - (1205)
Power, Clock, Thermal, and Reset Concerns - (1206)
Debug and Test Problems - (1207)
Cross-Cutting Problems - (1208)
Physical Access Issues and Concerns - (1388)
Not View-1194
View-1003 not View-1194
Memory related – Allocate
Memory related – Deallocate
Memory related – Initialize Object
Memory related – Dereference
Memory related – Read
Memory related – Write
CWEs by Abstraction:
Pillar
Class
Base
Variant
CWEs Relation by Nature:
ChildOf
CWEs Relation by View:
View-1000 and View-1194
Only View-1194
Only View-1000
CWE28F Tool, I. Bojanova

Fig. 24. HW CWE complete graph with memory-related weaknesses highlighted

7. Software Assurance Trends Categories

In addition to the views previously presented, there is a Software Development view (CWE-699). Figure 25 shows the View-699 CWEs that overlap with the complete HW CWEs graph (from Fig. 1).

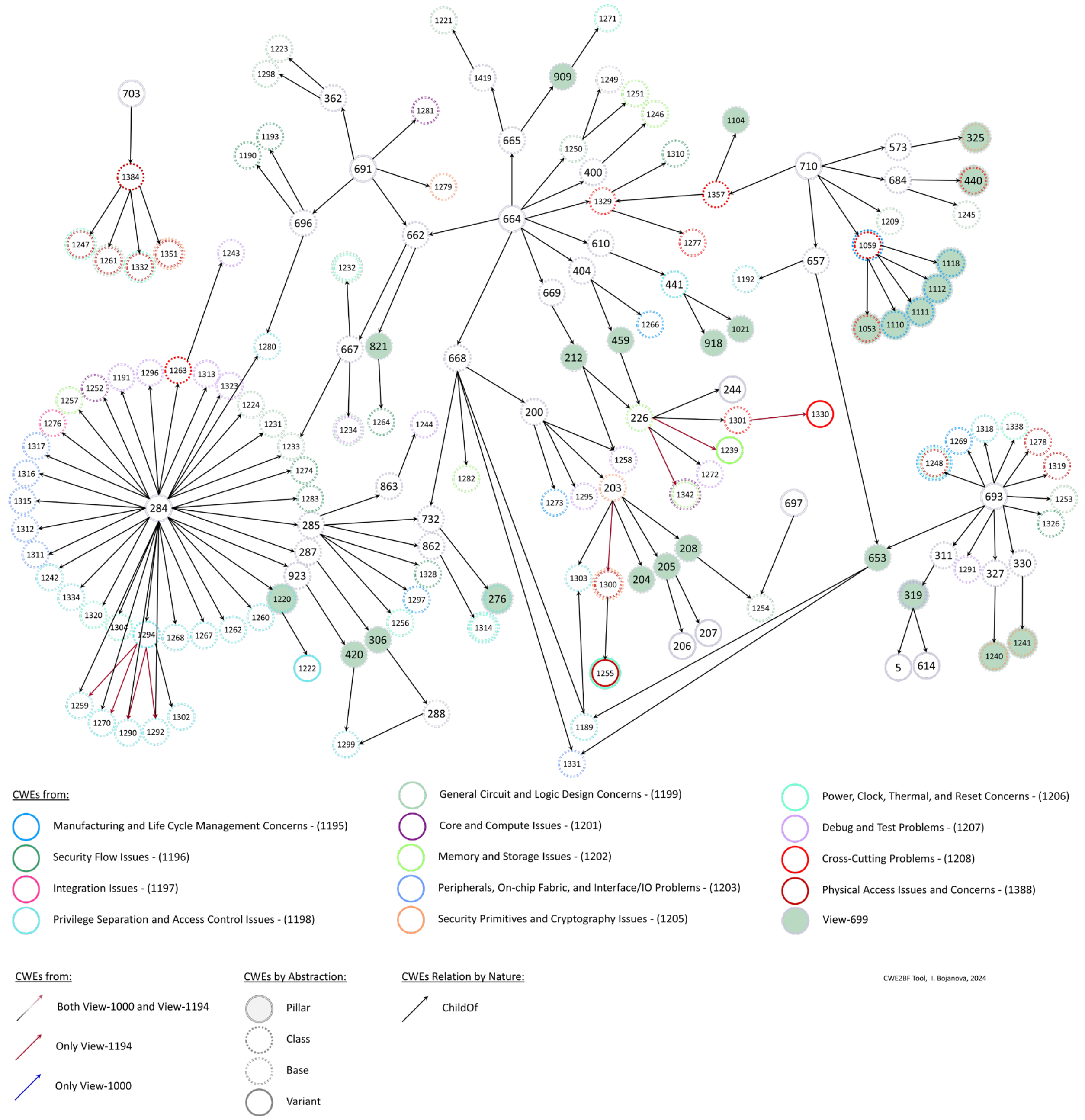

CWEs from:
Manufacturing and Life Cycle Management Concerns - (1195)
Security Flow Issues - (1196)
Integration Issues - (1197)
Privilege Separation and Access Control Issues - (1198)
General Circuit and Logic Design Concerns - (1199)
Core and Compute Issues - (1201)
Memory and Storage Issues - (1202)
Peripherals, On-chip Fabric, and Interface/IO Problems - (1203)
Security Primitives and Cryptography Issues - (1205)
Power, Clock, Thermal, and Reset Concerns - (1206)
Debug and Test Problems - (1207)
Cross-Cutting Problems - (1208)
Physical Access Issues and Concerns - (1388)
View-699
CWEs from:
Both View-1000 and View-1194
Only View-1194
Only View-1000
CWEs by Abstraction:
Pillar
Class
Base
Variant
CWEs Relation by Nature:
ChildOf
CWE2BF Tool, I. Bojanova, 2024

Fig. 25. View-699 CWEs that overlap with View-1194 highlighted

Only 12 CWEs are both in View-1194 and View-699. Organized by the View-699 categories, they are:

CWE View-699> CWE Category: Permission Issues – (CWE-275)

- CWE-276: Incorrect Default Permissions

CWE View-699> CWE Category: Cryptographic Issues – (CWE-310)

- CWE-325: Missing Cryptographic Step

CWE View-699> CWE Category: Behavioral Problems – (CWE-438)

- CWE-440: Expected Behavior Violation

CWE View-699> CWE Category: Documentation Issues – (CWE-1125)

- CWE-1053: Missing Documentation for Design
- CWE-1110: Incomplete Design Documentation
- CWE-1111: Incomplete I/O Documentation
- CWE-1112: Incomplete Documentation of Program Execution
- CWE-1118: Insufficient Documentation of Error Handling Techniques

CWE View-699> CWE Category: Authorization Errors – (CWE-1212)

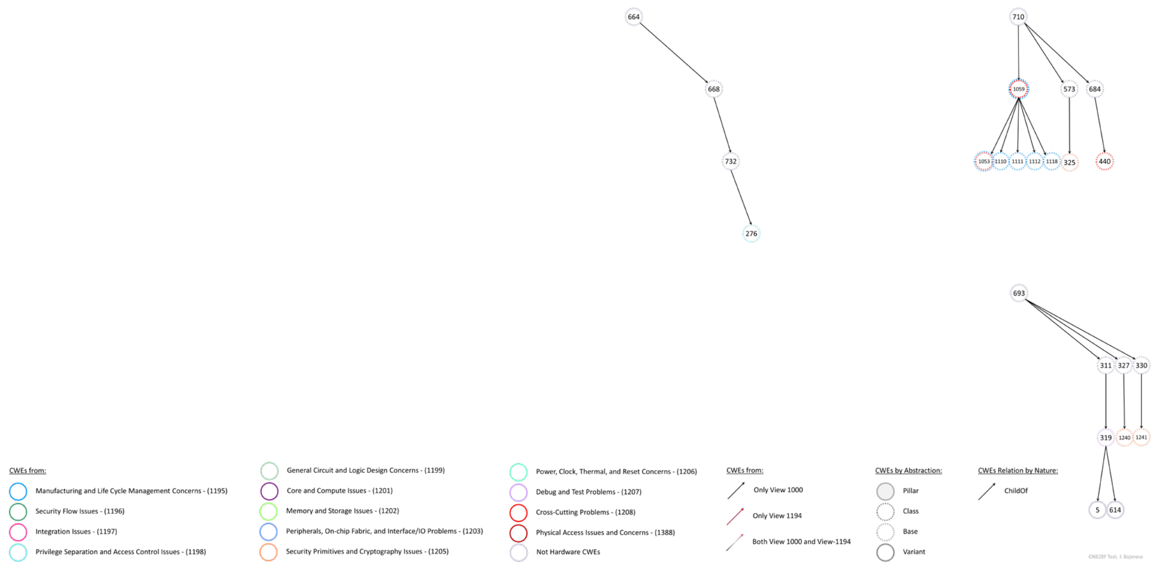
- CWE-1220: Insufficient Granularity of Access Control

CWE View-699> CWE Category: Information Management Errors – (CWE-199)

- CWE-319: Cleartext Transmission of Sensitive Information
- CWE-1240: Use of a Cryptographic Primitive with a Risky Implementation
- CWE-1241: Use of Predictable Algorithm in Random Number Generator

Figure 26 provides a separate view of these 12 CWEs.

983


664
668
732
276
710
1059
573
684
1053
1110
1111
1112
1118
325
440
693
311
327
330
319
1240
1241
5
614
CWEs from:
Manufacturing and Life Cycle Management Concerns - (1195)
Security Flow Issues - (1196)
Integration Issues - (1197)
Privilege Separation and Access Control Issues - (1198)
General Circuit and Logic Design Concerns - (1199)
Core and Compute Issues - (1201)
Memory and Storage Issues - (1202)
Peripherals, On-chip Fabric, and Interface/IO Problems - (1203)
Security Primitives and Cryptography Issues - (1205)
Power, Clock, Thermal, and Reset Concerns - (1206)
Debug and Test Problems - (1207)
Cross-Cutting Problems - (1208)
Physical Access Issues and Concerns - (1388)
Not Hardware CWEs
CWEs from:
Only View 1000
Only View 1194
Both View 1000 and View-1194
CWEs by Abstraction:
Pillar
Class
Base
Variant
CWEs Relation by Nature:
ChildOf
CWE2BF Tool, I. Bojanova

984

985 **Fig. 26. The 12 CWEs in both View-1194 and View-699**

986

8. Conclusion

Historically held notions that hardware is invulnerable have been shown to be incorrect. This work has presented 98 hardware security failure scenarios that demonstrate **what** an attacker can do, **where** can they do it, and **how** can they do it. Each scenario describes a type of vulnerability that can be instantiated in many different ways on distinct hardware platforms. Almost all of these scenarios represent significant security concerns.

However, there are few known HW vulnerabilities. As of February 22, 2024, there were only 131 HW CVEs. This can be partially explained by HW developers finding and removing HW vulnerabilities during the design process, meaning that they are never added to the CVE. At the same time, the number of HW CVEs may be artificially low because HW developers are reticent to acknowledge vulnerabilities in shipped products due to the inability to resolve or mitigate them. It is also possible that the restricted programming languages used for HW design limit the possibility of introducing vulnerabilities relative to more general software programming languages. Another factor could be that HW security has only recently received significantly heightened attention from the security community.

Hardware is a new focal point in the unending conflict between computer security hackers and defenders. Vulnerabilities can have serious consequences because of the large deployed base of chips and the inability to fix vulnerabilities on those chips. There are many ways in which HW can fail from a security perspective, and there is ample justification for securing HW infrastructure. HW is the foundation of computing and must be trustworthy.

References

The references are organized into general references and CWE references.

General References

[1] Bellay, Forte, Taylor (2021) *Hardware Vulnerability Description, Sharing and Reporting: Challenges and Opportunities*, Available at https://dforte.ece.ufl.edu/wp-content/uploads/sites/65/2021/05/GOMACTech_conf.pdf

[2] McConnell (2004) *Code Complete: A Practical Handbook of Software Construction*, Second Edition, Available at https://people.engr.tamu.edu/slupoli/notes/ProgrammingStudio/supplements/Code%20Complete%202nd.pdf

[3] Bojanova, Irena, et al. 'Bug, fault, error, or weakness: Demystifying software security vulnerabilities.' IT Professional 25.01 (2023): 7-12. Available at https://ieeexplore.ieee.org/document/10077830

[4] MITRE (2024) *CWE/CAPEC Board*, Available at https://cwe.mitre.org/community/board.html

[5] HW CWE SIG (2024) Hardware CWE Special Interest Group – Mission and Initial Guidance, Available at https://cwe.mitre.org/documents/HW_CWE_SIG.pdf

[6] MITRE (2024) *CVE*. Available at http://cve.mitre.org

[7] NIST (2024) *National Vulnerability Database*. Available at https://nvd.nist.gov

[8] MITRE (2024) *New to CWE*. Available at https://cwe.mitre.org/about/new_to_cwe.html

[9] MITRE (2024) *CWE Common Weakness Enumeration*. Available at https://cwe.mitre.org/index.html

CWE References

[CWE-203] Preliminary List Of Vulnerability Examples for Researchers (PLOVER) Project Team (2006) CWE-203: Observable Discrepancy. (The MITRE Corporation). Submission date 2006-07-19. Available at https://cwe.mitre.org/data/definitions/203.html

[CWE-226] PLOVER Project Team (2006) CWE-226: Sensitive Information in Resource Not Removed Before Reuse. (The MITRE Corporation). Submission date 2006-07-19. Available at https://cwe.mitre.org/data/definitions/226.html

[CWE-276] PLOVER Project Team (2006) CWE-276: Incorrect Default Permissions. (The MITRE Corporation). Submission date 2006-07-19. Available at https://cwe.mitre.org/data/definitions/276.html

[CWE-319] PLOVER Project Team (2006) CWE-319: Cleartext Transmission of Sensitive Information. (The MITRE Corporation). Submission date 2006-07-19. Available at https://cwe.mitre.org/data/definitions/319.html

[CWE-325] PLOVER Project Team (2006) CWE-325: Missing Cryptographic Step. (The MITRE Corporation). Submission date 2006-07-19. Available at https://cwe.mitre.org/data/definitions/325.html

[CWE-440] PLOVER Project Team (2006) CWE-440: Expected Behavior Violation. (The MITRE Corporation). Submission date 2006-07-19. Available at https://cwe.mitre.org/data/definitions/440.html

[CWE-441] PLOVER Project Team (2006) CWE-441: Unintended Proxy or Intermediary ('Confused Deputy'). (The MITRE Corporation). Submission date 2006-07-19. Available at https://cwe.mitre.org/data/definitions/441.html

[CWE-1053] CWE Content Team (2019) CWE-1053: Missing Documentation for Design. (The MITRE Corporation). Submission date 2019-01-03. Available at https://cwe.mitre.org/data/definitions/1053.html

[CWE-1059] CWE Content Team (2019) CWE-1059: Insufficient Technical Documentation. (The MITRE Corporation). Submission date 2019-01-03. Available at https://cwe.mitre.org/data/definitions/1059.html

[CWE-1189] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1189: Improper Isolation of Shared Resources on System-on-a-Chip (SoC). (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1189.html

[CWE-1190] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1190: DMA Device Enabled Too Early in Boot Phase. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1190.html

[CWE-1191] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1191: On-Chip Debug and Test Interface With Improper Access Control. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1191.html

[CWE-1192] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1192: Improper Identifier for IP Block used in System-On-Chip (SOC). (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1192.html

[CWE-1193] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1193: Power-On of Untrusted Execution Core Before Enabling Fabric Access Control. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1193.html

[CWE-1209] Sherman B (2020) CWE-1209: Failure to Disable Reserved Bits. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1209.html

[CWE-1220] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1220: Insufficient Granularity of Access Control. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1220.html

[CWE-1221] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1221: Incorrect Register Defaults or Module Parameters. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1221.html

[CWE-1222] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1222: Insufficient Granularity of Address Regions Protected by Register Locks. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1222.html

[CWE-1223] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1223: Race Condition for Write-Once Attributes. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1223.html

[CWE-1224] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1224: Improper Restriction of Write-Once Bit Fields. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1224.html

[CWE-1231] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1231: Improper Prevention of Lock Bit Modification. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1231.html

[CWE-1232] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1232: Improper Lock Behavior After Power State Transition. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1232.html

[CWE-1233] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1233: Security-Sensitive Hardware Controls with Missing Lock Bit Protection. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1233.html

[CWE-1234] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1234: Hardware Internal or Debug Modes Allow Override of Locks. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1234.html

[CWE-1239] Fern N (2020) CWE-1239: Improper Zeroization of Hardware Register. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1239.html

[CWE-1240] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1240: Use of a Cryptographic Primitive with a Risky Implementation. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1240.html

[CWE-1241] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1241: Use of Predictable Algorithm in Random Number Generator. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1241.html

[CWE-1242] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1242: Inclusion of Undocumented Features or Chicken Bits. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1242.html

[CWE-1243] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1243: Sensitive Non-Volatile Information Not Protected During Debug. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1243.html

[CWE-1244] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1244: Internal Asset Exposed to Unsafe Debug Access Level or State. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1244.html

[CWE-1245] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1245: Improper Finite State Machines (FSMs) in Hardware Logic. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1245.html

[CWE-1246] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1246: Improper Write Handling in Limited-write Non-Volatile Memories. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1246.html

[CWE-1247] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1247: Improper Protection Against Voltage and Clock Glitches. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1247.html

[CWE-1248] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1248: Semiconductor Defects in Hardware Logic with Security-Sensitive Implications. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1248.html

[CWE-1250] CWE Content Team (2020) CWE-1250: Improper Preservation of Consistency Between Independent Representations of Shared State. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1250.html

[CWE-1251] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1251: Mirrored Regions with Different Values. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1251.html

[CWE-1252] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1252: CPU Hardware Not Configured to Support Exclusivity of Write and Execute Operations. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1252.html

[CWE-1253] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1253: Incorrect Selection of Fuse Values. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1253.html

[CWE-1254] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1254: Incorrect Comparison Logic Granularity. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1254.html

[CWE-1255] CWE Content Team (2020) CWE-1255: Comparison Logic is Vulnerable to Power Side-Channel Attacks. (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1255.html

[CWE-1256] Fern N (2020) CWE-1256: Improper Restriction of Software Interfaces to Hardware Features. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1256.html

[CWE-1257] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1257: Improper Access Control Applied to Mirrored or Aliased Memory Regions. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1257.html

[CWE-1258] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1258: Exposure of Sensitive System Information Due to Uncleared Debug Information. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1258.html

[CWE-1259] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1259: Improper Restriction of Security Token Assignment. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1259.html

[CWE-1260] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1260: Improper Handling of Overlap Between Protected Memory Ranges. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1260.html

[CWE-1261] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1261: Improper Handling of Single Event Upsets. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1261.html

[CWE-1262] Fern N (2020) CWE-1262: Improper Access Control for Register Interface. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1262.html

[CWE-1263] CWE Content Team (2020) CWE-1263: Improper Physical Access Control. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1263.html

[CWE-1264] Fern N (2020) CWE-1264: Hardware Logic with Insecure De-Synchronization between Control and Data Channels. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1264.html

[CWE-1266] Wortman PA (2020) CWE-1266: Improper Scrubbing of Sensitive Data from Decommissioned Device. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1266.html

[CWE-1267] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1267: Policy Uses Obsolete Encoding. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1267.html

[CWE-1268] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1268: Policy Privileges are not Assigned Consistently Between Control and Data Agents. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1268.html

[CWE-1269] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1269: Product Released in Non-Release Configuration. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1269.html

[CWE-1270] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1270: Generation of Incorrect Security Tokens. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1270.html

[CWE-1271] Fern N (2020) CWE-1271: Uninitialized Value on Reset for Registers Holding Security Settings. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1271.html

[CWE-1272] Manna PK, Khattri H, Kanuparthi A (2020) CWE-1272: Sensitive Information Uncleared Before Debug/Power State Transition. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1272.html

[CWE-1273] Manna PK, Khattri H, Kanuparthi A (2020) CWE-1273: Device Unlock Credential Sharing. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1273.html

[CWE-1274] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1274: Improper Access Control for Volatile Memory Containing Boot Code. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1274.html

[CWE-1276] Fern N (2020) CWE-1276: Hardware Child Block Incorrectly Connected to Parent System. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1276.html

[CWE-1277] Wortman PA (2020) CWE-1277: Firmware Not Updateable. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1277.html

[CWE-1278] Fern N (2020) CWE-1278: Missing Protection Against Hardware Reverse Engineering Using Integrated Circuit (IC) Imaging Techniques. (The MITRE

Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1278.html

[CWE-1279] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1279: Cryptographic Operations are run Before Supporting Units are Ready. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1279.html

[CWE-1280] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1280: Access Control Check Implemented After Asset is Accessed. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1280.html

[CWE-1281] Fern N (2020) CWE-1281: Sequence of Processor Instructions Leads to Unexpected Behavior. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1281.html

[CWE-1282] Fern N (2020) CWE-1282: Assumed-Immutable Data is Stored in Writable Memory. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1282.html

[CWE-1283] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1283: Mutable Attestation or Measurement Reporting Data. (The MITRE Corporation). Submission date 2020-02-24. Available at https://cwe.mitre.org/data/definitions/1283.html

[CWE-1290] Kanuparthi A, Khattri H, Manna PK (2020) CWE-1290: Incorrect Decoding of Security Identifiers . (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1290.html

[CWE-1291] Manna PK, Khattri H, Kanuparthi A (2020) CWE-1291: Public Key Re-Use for Signing both Debug and Production Code. (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1291.html

[CWE-1292] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1292: Incorrect Conversion of Security Identifiers. (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1292.html

[CWE-1294] CWE Content Team (2020) CWE-1294: Insecure Security Identifier Mechanism. (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1294.html

[CWE-1295] Manna PK, Khattri H, Kanuparthi A (2020) CWE-1295: Debug Messages Revealing Unnecessary Information. (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1295.html

[CWE-1296] Kanuparthi A, Khattri H, Manna PK (2020) CWE-1296: Incorrect Chaining or Granularity of Debug Components. (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1296.html

[CWE-1297] Kanuparthi A, Khattri H, Manna PK (2020) CWE-1297: Unprotected Confidential Information on Device is Accessible by OSAT Vendors. (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1297.html

[CWE-1298] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1298: Hardware Logic Contains Race Conditions. (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1298.html

[CWE-1299] Kanuparthi A, Khattri H, Manna PK, Mangipudi NKV (2020) CWE-1299: Missing Protection Mechanism for Alternate Hardware Interface. (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1299.html

[CWE-1300] Fern N (2020) CWE-1300: Improper Protection of Physical Side Channels. (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1300.html

[CWE-1301] Fern N (2020) CWE-1301: Insufficient or Incomplete Data Removal within Hardware Component. (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1301.html

[CWE-1302] Kanuparthi A, Khattri H, Manna PK (2020) CWE-1302: Missing Source Identifier in Entity Transactions on a System-On-Chip (SOC). (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1302.html

[CWE-1303] Fern N (2020) CWE-1303: Non-Transparent Sharing of Microarchitectural Resources. (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1303.html

[CWE-1304] Accellera Systems Initiative (2020) CWE-1304: Improperly Preserved Integrity of Hardware Configuration State During a Power Save/Restore Operation. (The MITRE Corporation). Submission date 2020-08-20. Available at https://cwe.mitre.org/data/definitions/1304.html

[CWE-1310] Mangipudi NKV (2020) CWE-1310: Missing Ability to Patch ROM Code. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1310.html

[CWE-1311] Kanuparthi A, Khattri H, Manna P (2020) CWE-1311: Improper Translation of Security Attributes by Fabric Bridge. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1311.html

[CWE-1312] Kanuparthi A, Khattri H, Manna PK (2020) CWE-1312: Missing Protection for Mirrored Regions in On-Chip Fabric Firewall. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1312.html

[CWE-1313] Sherman B (2020) CWE-1313: Hardware Allows Activation of Test or Debug Logic at Runtime. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1313.html

[CWE-1314] Khattri H, Manna PK, Kanuparthi AA (2020) CWE-1314: Missing Write Protection for Parametric Data Values. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1314.html

[CWE-1315] Kanuparthi A, Khattri H, Manna PK (2020) CWE-1315: Improper Setting of Bus Controlling Capability in Fabric End-point. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1315.html

[CWE-1316] Kanuparthi A, Khattri H, Manna PK (2020) CWE-1316: Fabric-Address Map Allows Programming of Unwarranted Overlaps of Protected and Unprotected Ranges. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1316.html

[CWE-1317] Kanuparthi A, Khattri H, Manna PK (2020) CWE-1317: Improper Access Control in Fabric Bridge. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1317.html

[CWE-1318] Kanuparthi A, Khattri H, Manna PK (2020) CWE-1318: Missing Support for Security Features in On-chip Fabrics or Buses. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1318.html

[CWE-1319] Leger S, Narasipur R (2020) CWE-1319: Improper Protection against Electromagnetic Fault Injection (EM-FI). (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1319.html

[CWE-1320] Khattri H, Kanuparthi A, Manna PK (2020) CWE-1320: Improper Protection for Outbound Error Messages and Alert Signals. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1320.html

[CWE-1323] Khattri H, Manna PK, Kanuparthi AA (2020) CWE-1323: Improper Management of Sensitive Trace Data. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1323.html

[CWE-1326] Kanuparthi A, Khattri H, Manna PK (2020) CWE-1326: Missing Immutable Root of Trust in Hardware. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1326.html

[CWE-1328] Kanuparthi A, Khattri H, Manna PK (2020) CWE-1328: Security Version Number Mutable to Older Versions. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1328.html

[CWE-1329] CWE Content Team (2020) CWE-1329: Reliance on Component That is Not Updateable. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1329.html

[CWE-1330] Khattri H, Kanuparthi A, Manna PK (2020) CWE-1330: Remanent Data Readable after Memory Erase. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1330.html

[CWE-1331] Kanuparthi A, Khattri H, Manna PK (2020) CWE-1331: Improper Isolation of Shared Resources in Network On Chip (NoC). (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1331.html

[CWE-1332] Woudenberg J (2020) CWE-1332: Improper Handling of Faults that Lead to Instruction Skips. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1332.html

[CWE-1334] Pangburn J (2020) CWE-1334: Unauthorized Error Injection Can Degrade Hardware Redundancy. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1334.html

[CWE-1338] Kanuparthi A, Khattri H, Manna PK (2020) CWE-1338: Improper Protections Against Hardware Overheating. (The MITRE Corporation). Submission date 2020-12-10. Available at https://cwe.mitre.org/data/definitions/1338.html

[CWE-1342] Nordstrom A, Althoff A (2021) CWE-1342: Information Exposure through Microarchitectural State after Transient Execution. (The MITRE Corporation). Submission date 2021-10-28. Available at https://cwe.mitre.org/data/definitions/1342.html

[CWE-1351] Wortman PA (2021) CWE-1351: Improper Handling of Hardware Behavior in Exceptionally Cold Environments. (The MITRE Corporation). Submission date 2021-07-20. Available at https://cwe.mitre.org/data/definitions/1351.html

[CWE-1357] CWE Content Team (2022) CWE-1357: Reliance on Insufficiently Trustworthy Component. (The MITRE Corporation). Submission date 2022-04-28. Available at https://cwe.mitre.org/data/definitions/1357.html

[CWE-1384] CWE Content Team (2022) CWE-1384: Improper Handling of Physical or Environmental Conditions. (The MITRE Corporation). Submission date 2022-04-28. Available at https://cwe.mitre.org/data/definitions/1384.html

[CWE-1420] Constable SD (2024) CWE-1420: Exposure of Sensitive Information during Transient Execution. (The MITRE Corporation). Submission date 2024-02-29. Available at https://cwe.mitre.org/data/definitions/1420.html

[CWE-1421] Constable SD (2024) CWE-1421: Exposure of Sensitive Information in Shared Microarchitectural Structures during Transient Execution. (The MITRE Corporation). Submission date 2024-02-29. Available at https://cwe.mitre.org/data/definitions/1421.html

[CWE-1422] Constable SD (2024) CWE-1422: Exposure of Sensitive Information caused by Incorrect Data Forwarding during Transient Execution. (The MITRE Corporation).

Submission date 2024-02-29. Available at https://cwe.mitre.org/data/definitions/1422.html

[CWE-1423] Constable SD (2024) CWE-1423: Exposure of Sensitive Information caused by Shared Microarchitectural Predictor State that Influences Transient Execution. (The MITRE Corporation). Submission date 2024-02-29. Available at https://cwe.mitre.org/data/definitions/1423.html

Appendix A. List of Symbols, Abbreviations, and Acronyms

CPU
Central Processing Unit

DoS
Denial of Service

FSM
Finite-State Machine

IP
Intellectual Property

JTAG
Joint Test Action Group

MMU
Memory Management Unit

MPU
Memory Protection Unit

NoC
Network-on-Chip

NVM
Non-Volatile Memory

OS
Operating System

OTP
One-Time Programmable Memory

ROM
Read-Only Memory

SEU
Single-Event Upset

SoC
System-on-a-Chip

TAP
Test Access Port

VM
Volatile Memory

Appendix B. Analysis of the Complete Hardware Weakness Graph

Figure 1 in Sec. 3.2 shows the complete HW CWE graph. The root nodes are the seven HW applicable pillars under the Research Concepts view. Table 1 provides statistics on the types of CWEs in the graph.

Table 1. Statistics on the complete HW CWE graph

	Non-HW CWEs	HW CWEs	All CWEs
All	50	108	158
Pillar	7	0	7
Class	25	6	31
Base	13	98	111
Variant	5	4	9
Compound	0	0	0

To construct an HW CWE graph, create a directed graph for all CWEs using the relationships provided by the Research Concepts view (CWE-1000). Remove all nodes that are unreachable from any of the seven HW applicable pillars as well as all nodes without at least one HW CWE as a descendant, unless they themselves are HW CWEs. Add in any edges from the Hardware Design view (CWE-1194) that are not already in the graph.

B.1. Hardware Design Category Overlay

In Fig. 1, nodes with more than one outline belong to more than one HW design category. There are four CWEs that belong to three categories: CWE-1248 to CWE-1195, CWE-1206, and CWE-1388 and CWEs-1421, 1422, and 1423 to CWE-1198, CWE-1201, and CWE-1202. There are 12 CWEs that belong to two categories: CWE-1247, CWE-1255, and CWE-1332 to CWE-1206 and CWE-1388; CWE-1300 and CWE-1351 to CWE-1205 and CWE-1388; CWE-1059 to CWE-1195 and CWE-1208; CWE-1232 to CWE-1199 and CWE-1206; CWE-1234 to CWE-1199 and CWE-1207; CWE-1261 to CWE-1199 and CWE-1388; CWE-1314 to CWE-1198 and CWE-1206; CWE-1342 and CWE-1420 to CWE-1201 and CWE-1202; and CWE-1351 to CWE-1205 and CWE-1388.

Table 2. Mapping of HW CWEs to HW Categories

CWE\Category	CWE-1195	CWE-1198	CWE-1199	CWE-1201	CWE-1202	CWE-1205	CWE-1206	CWE-1207	CWE-1208	CWE-1388
CWE-1248	✓						✓			✓
CWE-1247							✓			✓
CWE-1255							✓			✓
CWE-1332							✓			✓
CWE-1300						✓				✓
CWE-1351						✓				✓
CWE-1059	✓								✓	
CWE-1232			✓				✓			
CWE-1234			✓					✓		
CWE-1261			✓							✓
CWE-1314		✓					✓			
CWE-1342				✓	✓					
CWE-1420				✓	✓					
C WE-1421		✓		✓	✓					
C WE-1422		✓		✓	✓					
C WE-1423		✓		✓	✓					

B.2. Comparison of View-1000 and View-1194 Relationships

There are seven relationships that belong to both View-1000 and View-1194 depicted on the digraph with gradient black-to-red edges (arrows): CWE-226→CWE-1342, CWE-226→CWE-1239, CWE-1301→CWE-CWE-1330, CWE-203→CWE-CWE-1300, CWE-1420→ CWE-1421, CWE-1422, and CWE-1423. Four other relationships belong only to view 1194 and are depicted with red edges (arrows): CWE-1294→CWE-1259, CWE-1294→1270, CWE-1294→ CWE-1290, and CWE-1294→CWE-1292. The rest of the relations only belongto View-1000 and are depicted in black.

The following parent-child relations are only present in View-1000, but both of their nodes pertain to View-1194 as well: CWE-1220→CWE-1222; CWE-1263→CWE-1243; CWE-1294→CWE-1302; CWE-1384→CWEs-1247, 1261, 1332, and 1351; CWE-226→CWEs-1272 and

1301; CWE-203→CWE-1303; CWE-1300→CWE-1255; CWE-1357→CWEs-1329; CWE-1329→1277 and 1310; and CWE-1059→CWEs-1053, 1110, 1111, 1112, and 1118.

CWE-208 is only used in View-1194 as a intermediary, but both its parent and child pertain to View-1194: CWE-20→ CWE-208→ CWE-1254.

Appendix C. Weakness Hierarchy — Improper Access Control

The CWEs for this pillar are listed in a strict hierarchical tree structure to allow for easy perusal of all relevant CWEs. Some CWEs are duplicated because they appear under multiple classes within the same pillar. The full graph view in Fig. 1 shows the complex relationships between many of the HW CWEs.

Each CWE is labelled with its abstraction type — Pillar: P, Class: C, Base: B, or Variant: V. Those marked with * are HW CWEs.

CWE-284 P Improper Access Control

Figure 27 shows the relationship of CWEs to each other and various attributes of the CWEs (e.g., hardware category and CWE abstraction).

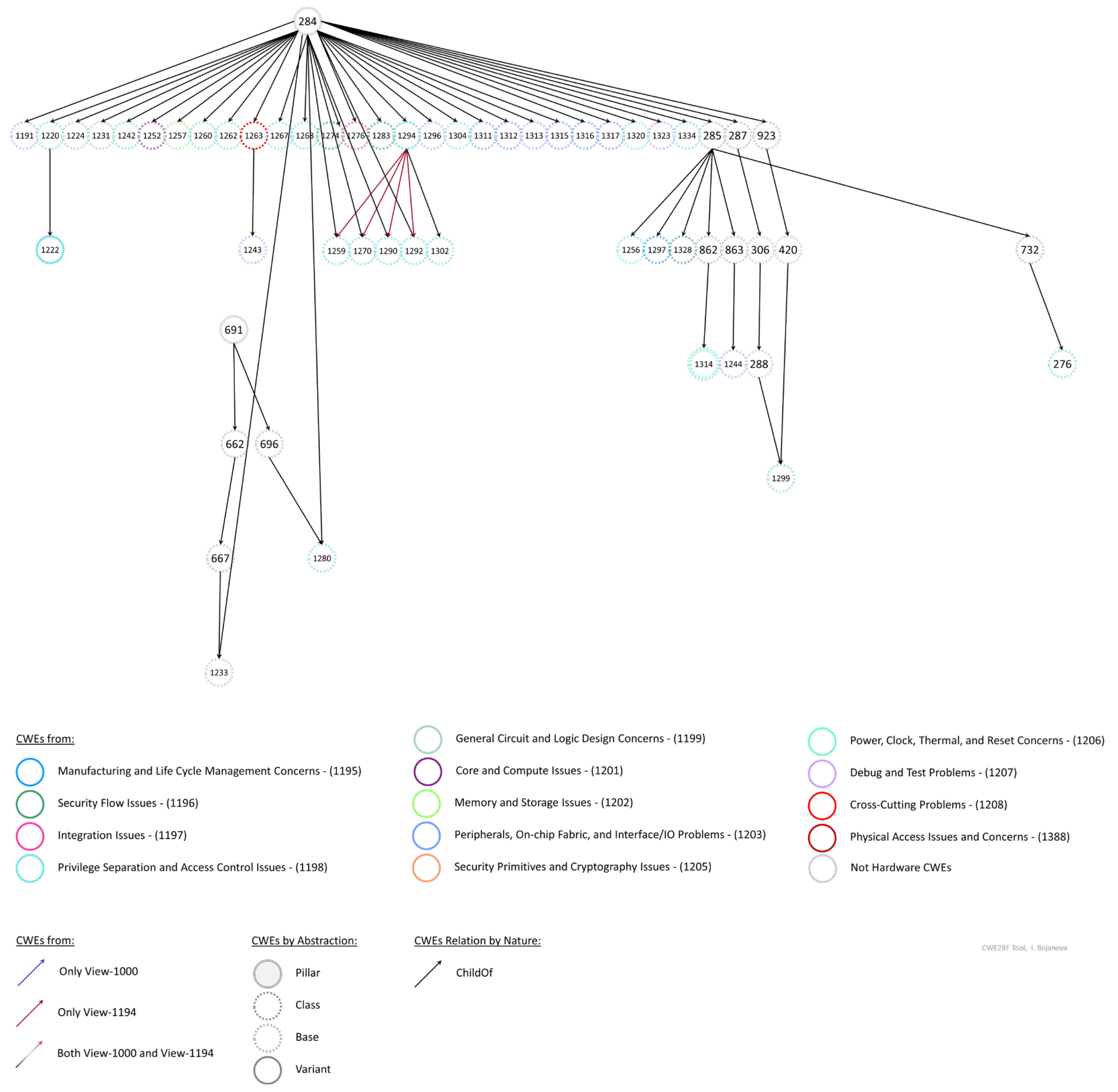

284
1191
1220
1224
1231
1242
1252
1257
1260
1262
1263
1267
1268
1274
1276
1283
1294
1296
1304
1311
1312
1313
1315
1316
1317
1320
1323
1334
285
287
923
1222
1243
1259
1270
1290
1292
1302
1256
1297
1328
862
863
306
420
732
691
1314
1244
288
276
662
696
1299
667
1280
1233
CWEs from:
Manufacturing and Life Cycle Management Concerns - (1195)
Security Flow Issues - (1196)
Integration Issues - (1197)
Privilege Separation and Access Control Issues - (1198)
General Circuit and Logic Design Concerns - (1199)
Core and Compute Issues - (1201)
Memory and Storage Issues - (1202)
Peripherals, On-chip Fabric, and Interface/IO Problems - (1203)
Security Primitives and Cryptography Issues - (1205)
Power, Clock, Thermal, and Reset Concerns - (1206)
Debug and Test Problems - (1207)
Cross-Cutting Problems - (1208)
Physical Access Issues and Concerns - (1388)
Not Hardware CWEs
CWEs from:
Only View-1000
Only View-1194
Both View-1000 and View-1194
CWEs by Abstraction:
Pillar
Class
Base
Variant
CWEs Relation by Nature:
ChildOf
CWE2BF Tool, I. Bojanova

Fig. 27. HW CWE Category Graph: Improper Access Control

- CWE-1191 B On-Chip Debug and Test Interface With Improper Access Control *
- CWE-1220 B Insufficient Granularity of Access Control *
  - CWE-1222 V Insufficient Granularity of Address Regions Protected by Register Locks *
- CWE-1224 B Improper Restriction of Write-Once Bit Fields *
- CWE-1231 B Improper Prevention of Lock Bit Modification *
- CWE-1233 B Security-Sensitive Hardware Controls with Missing Lock Bit Protection *
- CWE-1242 B Inclusion of Undocumented Features or Chicken Bits *

- CWE-1252 B CPU Hardware Not Configured to Support Exclusivity of Write and Execute Operations *
- CWE-1257 B Improper Access Control Applied to Mirrored or Aliased Memory Regions *
- CWE-1259 B Improper Restriction of Security Token Assignment *
- CWE-1260 B Improper Handling of Overlap Between Protected Memory Ranges *
- CWE-1262 B Improper Access Control for Register Interface *
- CWE-1263 C Improper Physical Access Control *
    - CWE-1243 B Sensitive Non-Volatile Information Not Protected During Debug *
- CWE-1267 B Policy Uses Obsolete Encoding *
- CWE-1268 B Policy Privileges are not Assigned Consistently Between Control and Data Agents *
- CWE-1270 B Generation of Incorrect Security Tokens *
- CWE-1274 B Improper Access Control for Volatile Memory Containing Boot Code *
- CWE-1276 B Hardware Child Block Incorrectly Connected to Parent System *
- CWE-1280 B Access Control Check Implemented After Asset is Accessed *
- CWE-1283 B Mutable Attestation or Measurement Reporting Data *
- CWE-1290 B Incorrect Decoding of Security Identifiers *
- CWE-1292 B Incorrect Conversion of Security Identifiers *
- CWE-1294 C Insecure Security Identifier Mechanism *
    - CWE-1302 B Missing Security Identifier *
- CWE-1296 B Incorrect Chaining or Granularity of Debug Components *
- CWE-1304 B Improperly Preserved Integrity of Hardware Configuration State During a Power Save/Restore Operation *
- CWE-1311 B Improper Translation of Security Attributes by Fabric Bridge *
- CWE-1312 B Missing Protection for Mirrored Regions in On-Chip Fabric Firewall *
- CWE-1313 B Hardware Allows Activation of Test or Debug Logic at Runtime *
- CWE-1315 B Improper Setting of Bus Controlling Capability in Fabric End-point *
- CWE-1316 B Fabric-Address Map Allows Programming of Unwarranted Overlaps of Protected and Unprotected Ranges *
- CWE-1317 B Improper Access Control in Fabric Bridge *
- CWE-1320 B Improper Protection for Outbound Error Messages and Alert Signals *
- CWE-1323 B Improper Management of Sensitive Trace Data *

- CWE-1334 B Unauthorized Error Injection Can Degrade Hardware Redundancy *
- CWE-285 C Improper Authorization
    - CWE-1256 B Improper Restriction of Software Interfaces to Hardware Features *
    - CWE-1297 B Unprotected Confidential Information on Device is Accessible by OSAT Vendors *
    - CWE-1328 B Security Version Number Mutable to Older Versions *
    - CWE-732 C Incorrect Permission Assignment for Critical Resource
        - CWE-276 B Incorrect Default Permissions *
    - CWE-862 C Missing Authorization
        - CWE-1314 B Missing Write Protection for Parametric Data Values *
    - CWE-863 C Incorrect Authorization
        - CWE-1244 B Internal Asset Exposed to Unsafe Debug Access Level or State *
- CWE-287 C Improper Authentication
    - CWE-306 B Missing Authentication for Critical Function
        - CWE-288 B Authentication Bypass Using an Alternate Path or Channel
            - CWE-1299 B Missing Protection Mechanism for Alternate Hardware Interface *
- CWE-923 C Improper Restriction of Communication Channel to Intended Endpoints
    - CWE-420 B Unprotected Alternate Channel
        - CWE-1299 B Missing Protection Mechanism for Alternate Hardware Interface *

Appendix D. Weakness Hierarchy — Improper Adherence to Coding Standards

The CWEs for this pillar are listed in a strict hierarchical tree structure to allow for easy perusal of all relevant CWEs. Some CWEs are duplicated because they appear under multiple classes within the same pillar. The full graph view in Fig. 1 shows the complex relationships between many of the HW CWEs.

Each CWE is labelled with its abstraction type — Pillar: P, Class: C, Base: B, or Variant: V. Those marked with * are HW CWEs.

<u>CWE-710 P Improper Adherence to Coding Standards</u>

- CWE-1059 C Insufficient Technical Documentation *
    - CWE-1053 B Missing Documentation for Design *
- CWE-1209 B Failure to Disable Reserved Bits *
- CWE-1357 C Reliance on Insufficiently Trustworthy Component *
    - CWE-1329 B Reliance on Component That is Not Updateable *
        - CWE-1277 B Firmware Not Updateable *
        - CWE-1310 B Missing Ability to Patch ROM Code *
- CWE-573 C Improper Following of Specification by Caller
    - CWE-325 B Missing Cryptographic Step *
- CWE-657 C Violation of Secure Design Principles
    - CWE-1192 B System-on-Chip (SoC) Using Components without Unique, Immutable Identifiers *
    - CWE-653 B Improper Isolation or Compartmentalization
        - CWE-1189 B Improper Isolation of Shared Resources on System-on-a-Chip (SoC) *
            - CWE-1303 B Non-Transparent Sharing of Microarchitectural Resources *
        - CWE-1331 B Improper Isolation of Shared Resources in Network On Chip (NoC) *
- CWE-684 C Incorrect Provision of Specified Functionality
    - CWE-1245 B Improper Finite State Machines (FSMs) in Hardware Logic *
    - CWE-440 B Expected Behavior Violation *

Appendix E. Weakness Hierarchy — Improper Check or Handling of Exceptional Conditions

The CWEs for this pillar are listed in a strict hierarchical tree structure to allow for easy perusal of all relevant CWEs. Some CWEs are duplicated because they appear under multiple classes within the same pillar. The full graph view in Fig. 1 shows the complex relationships between many of the HW CWEs.

Each CWE is labelled with its abstraction type — Pillar: P, Class: C, Base: B, or Variant: V. Those marked with * are HW CWEs.

<u>CWE-703 P Improper Check or Handling of Exceptional Conditions</u>

- CWE-1384 C Improper Handling of Physical or Environmental Conditions *
    - CWE-1247 B Improper Protection Against Voltage and Clock Glitches *
    - CWE-1261 B Improper Handling of Single Event Upsets *
    - CWE-1332 B Improper Handling of Faults that Lead to Instruction Skips *
    - CWE-1351 B Improper Handling of Hardware Behavior in Exceptionally Cold Environments *

Appendix F. Weakness Hierarchy — Improper Control of a Resource Through its Lifetime

The CWEs for this pillar are listed in a strict hierarchical tree structure to allow for easy perusal of all relevant CWEs. Some CWEs are duplicated because they appear under multiple classes within the same pillar. The full graph view in Fig. 1 shows the complex relationships between many of the HW CWEs.

Each CWE is labelled with its abstraction type — Pillar: P, Class: C, Base: B, or Variant: V. Those marked with * are HW CWEs.

<u>CWE-664 P Improper Control of a Resource Through its Lifetime</u>

- CWE-1250 B Improper Preservation of Consistency Between Independent Representations of Shared State *
    - CWE-1251 B Mirrored Regions with Different Values *
- CWE-1329 B Reliance on Component That is Not Updateable *
    - CWE-1277 B Firmware Not Updateable *
    - CWE-1310 B Missing Ability to Patch ROM Code *
- CWE-400 C Uncontrolled Resource Consumption
    - CWE-1246 B Improper Write Handling in Limited-write Non-Volatile Memories *
- CWE-404 C Improper Resource Shutdown or Release
    - CWE-1266 B Improper Scrubbing of Sensitive Data from Decommissioned Device *
    - CWE-459 B Incomplete Cleanup
        - CWE-226 B Sensitive Information in Resource Not Removed Before Reuse *
            - CWE-1239 V Improper Zeroization of Hardware Register *
            - CWE-1272 B Sensitive Information Uncleared Before Debug/Power State Transition *
            - CWE-1301 B Insufficient or Incomplete Data Removal within Hardware Component *
                - CWE-1330 V Remanent Data Readable after Memory Erase *
            - CWE-1342 B Information Exposure through Microarchitectural State after Transient Execution *
- CWE-610 C Externally Controlled Reference to a Resource in Another Sphere
    - CWE-441 C Unintended Proxy or Intermediary ('Confused Deputy') *
- CWE-662 C Improper Synchronization

    - CWE-667 C Improper Locking
        - CWE-1232 B Improper Lock Behavior After Power State Transition *
        - CWE-1233 B Security-Sensitive Hardware Controls with Missing Lock Bit Protection *
        - CWE-1234 B Hardware Internal or Debug Modes Allow Override of Locks *
    - CWE-821 B Incorrect Synchronization
        - CWE-1264 B Hardware Logic with Insecure De-Synchronization between Control and Data Channels *
- CWE-665 C Improper Initialization
    - CWE-1279 B Cryptographic Operations are run Before Supporting Units are Ready *
    - CWE-1419 C Incorrect Initialization of Resource
        - CWE-1221 B Incorrect Register Defaults or Module Parameters *
    - CWE-909 C Missing Initialization of Resource
        - CWE-1271 B Uninitialized Value on Reset for Registers Holding Security Settings *
- CWE-668 C Exposure of Resource to Wrong Sphere
    - CWE-1189 B Improper Isolation of Shared Resources on System-on-a-Chip (SoC) *
        - CWE-1303 B Non-Transparent Sharing of Microarchitectural Resources *
    - CWE-1282 B Assumed-Immutable Data is Stored in Writable Memory *
    - CWE-1331 B Improper Isolation of Shared Resources in Network On Chip (NoC) *
    - CWE-200 C Exposure of Sensitive Information to an Unauthorized Actor
        - CWE-1258 B Exposure of Sensitive System Information Due to Uncleared Debug Information *
        - CWE-1273 B Device Unlock Credential Sharing *
        - CWE-1295 B Debug Messages Revealing Unnecessary Information *
        - CWE-203 B Observable Discrepancy *
            - CWE-1300 B Improper Protection of Physical Side Channels *
                - CWE-1255 V Comparison Logic is Vulnerable to Power Side-Channel Attacks *
            - CWE-1303 B Non-Transparent Sharing of Microarchitectural Resources *

- CWE-208 B Observable Timing Discrepancy
  - CWE-1254 B Incorrect Comparison Logic Granularity *
- CWE-732 C Incorrect Permission Assignment for Critical Resource
  - CWE-276 B Incorrect Default Permissions *

- CWE-669 C Incorrect Resource Transfer Between Spheres
  - CWE-212 B Improper Removal of Sensitive Information Before Storage or Transfer
    - CWE-1258 B Exposure of Sensitive System Information Due to Uncleared Debug Information *
    - CWE-226 B Sensitive Information in Resource Not Removed Before Reuse *
      - CWE-1239 V Improper Zeroization of Hardware Register *
      - CWE-1272 B Sensitive Information Uncleared Before Debug/Power State Transition *
      - CWE-1301 B Insufficient or Incomplete Data Removal within Hardware Component *
        - CWE-1330 V Remanent Data Readable after Memory Erase *
      - CWE-1342 B Information Exposure through Microarchitectural State after Transient Execution *

Appendix G. Weakness Hierarchy — Incorrect Comparison

The CWEs for this pillar are listed in a strict hierarchical tree structure to allow for easy perusal of all relevant CWEs. Some CWEs are duplicated because they appear under multiple classes within the same pillar. The full graph view in Fig. 1 shows the complex relationships between many of the HW CWEs.

Each CWE is labelled with its abstraction type — Pillar: P, Class: C, Base: B, or Variant: V. Those marked with * are HW CWEs.

<u>CWE-697 P Incorrect Comparison</u>

- CWE-1254 B Incorrect Comparison Logic Granularity *

Appendix H. Weakness Hierarchy — Insufficient Control Flow Management

The CWEs for this pillar are listed in a strict hierarchical tree structure to allow for easy perusal of all relevant CWEs. Some CWEs are duplicated because they appear under multiple classes within the same pillar. The full graph view in Fig. 1 shows the complex relationships between many of the HW CWEs.

Each CWE is labelled with its abstraction type — Pillar: P, Class: C, Base: B, or Variant: V. Those marked with * are HW CWEs.

<u>CWE-691 P Insufficient Control Flow Management</u>

- CWE-1279 B Cryptographic Operations are run Before Supporting Units are Ready *
- CWE-1281 B Sequence of Processor Instructions Leads to Unexpected Behavior *
- CWE-362 C Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')
    - CWE-1223 B Race Condition for Write-Once Attributes *
    - CWE-1298 B Hardware Logic Contains Race Conditions *
- CWE-662 C Improper Synchronization
    - CWE-667 C Improper Locking
        - CWE-1232 B Improper Lock Behavior After Power State Transition *
        - CWE-1233 B Security-Sensitive Hardware Controls with Missing Lock Bit Protection *
        - CWE-1234 B Hardware Internal or Debug Modes Allow Override of Locks *
    - CWE-821 B Incorrect Synchronization
        - CWE-1264 B Hardware Logic with Insecure De-Synchronization between Control and Data Channels *
- CWE-696 C Incorrect Behavior Order
    - CWE-1190 B DMA Device Enabled Too Early in Boot Phase *
    - CWE-1193 B Power-On of Untrusted Execution Core Before Enabling Fabric Access Control *
    - CWE-1280 B Access Control Check Implemented After Asset is Accessed *

Appendix I. Weakness Hierarchy — Protection Mechanism Failure

The CWEs for this pillar are listed in a strict hierarchical tree structure to allow for easy perusal of all relevant CWEs. Some CWEs are duplicated because they appear under multiple classes within the same pillar. The full graph view in Fig. 1 shows the complex relationships between many of the HW CWEs.

Each CWE is labelled with its abstraction type — Pillar: P, Class: C, Base: B, or Variant: V. Those marked with * are HW CWEs.

CWE-693 P Protection Mechanism Failure

- CWE-1248 B Semiconductor Defects in Hardware Logic with Security-Sensitive Implications *
- CWE-1253 B Incorrect Selection of Fuse Values *
- CWE-1269 B Product Released in Non-Release Configuration *
- CWE-1278 B Missing Protection Against Hardware Reverse Engineering Using Integrated Circuit (IC) Imaging Techniques *
- CWE-1291 B Public Key Re-Use for Signing both Debug and Production Code *
- CWE-1318 B Missing Support for Security Features in On-chip Fabrics or Buses *
- CWE-1319 B Improper Protection against Electromagnetic Fault Injection (EM-FI) *
- CWE-1326 B Missing Immutable Root of Trust in Hardware *
- CWE-1338 B Improper Protections Against Hardware Overheating *
- CWE-311 C Missing Encryption of Sensitive Data
    - CWE-319 B Cleartext Transmission of Sensitive Information *
- CWE-327 C Use of a Broken or Risky Cryptographic Algorithm
    - CWE-1240 B Use of a Cryptographic Primitive with a Risky Implementation *
- CWE-330 C Use of Insufficiently Random Values
    - CWE-1241 B Use of Predictable Algorithm in Random Number Generator *
- CWE-653 B Improper Isolation or Compartmentalization
    - CWE-1189 B Improper Isolation of Shared Resources on System-on-a-Chip (SoC) *
        - CWE-1303 B Non-Transparent Sharing of Microarchitectural Resources *
    - CWE-1331 B Improper Isolation of Shared Resources in Network On Chip (NoC) *