Hardware CWE™
Special Interest Group (SIG)

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

MITRE

October 13, 2023

Agenda

REMINDER: This meeting is being recorded.

- **Housekeeping and Announcements**
- **Working Items for this meeting:**

1	CWE Nit Bits: Observed Examples	Bob H	5 min
2	Missing Initialization Weakness Sneak Preview (Discussion)	Steve C	15 min
3	Inclusive Language and HW CWEs	Steve C	15 min
4	Community Items	Bob H	15 min

Housekeeping

- **Schedule:**
  - **Next Meeting:**
    - **November 10th (US Federal Holiday) potentially rescheduled to November 3rd . December 8th follows.**
    - **12:30 – 1:30 PM EST (16:30 – 17:30 UTC)**
    - **Microsoft Teams**
- **Contact: cwe@mitre.org**
- **Mailing List:** *hw-cwe-special-interest-group-sig-list@mitre.org*
- **Minutes from previous meetings available on our GitHub site:**
  - **https://github.com/CWE-CAPEC/hw-cwe-sig**

Announcements

- **New CWE Content Development Repository (CDR) pilot now on GitHub! Currently invite only. Potential public release in November 2023.**

- **CWE 4.13 to be released October 26, 2023. Content freeze on October 20, 2023.**

- **Tentative: CISA strategy around Secure By Design/Secure By Default for Nov SIG**

- **HW CWE Spotlight: SIG Member to present internal tool developed that utilizes HW CWE – pushed from Oct to Nov.**

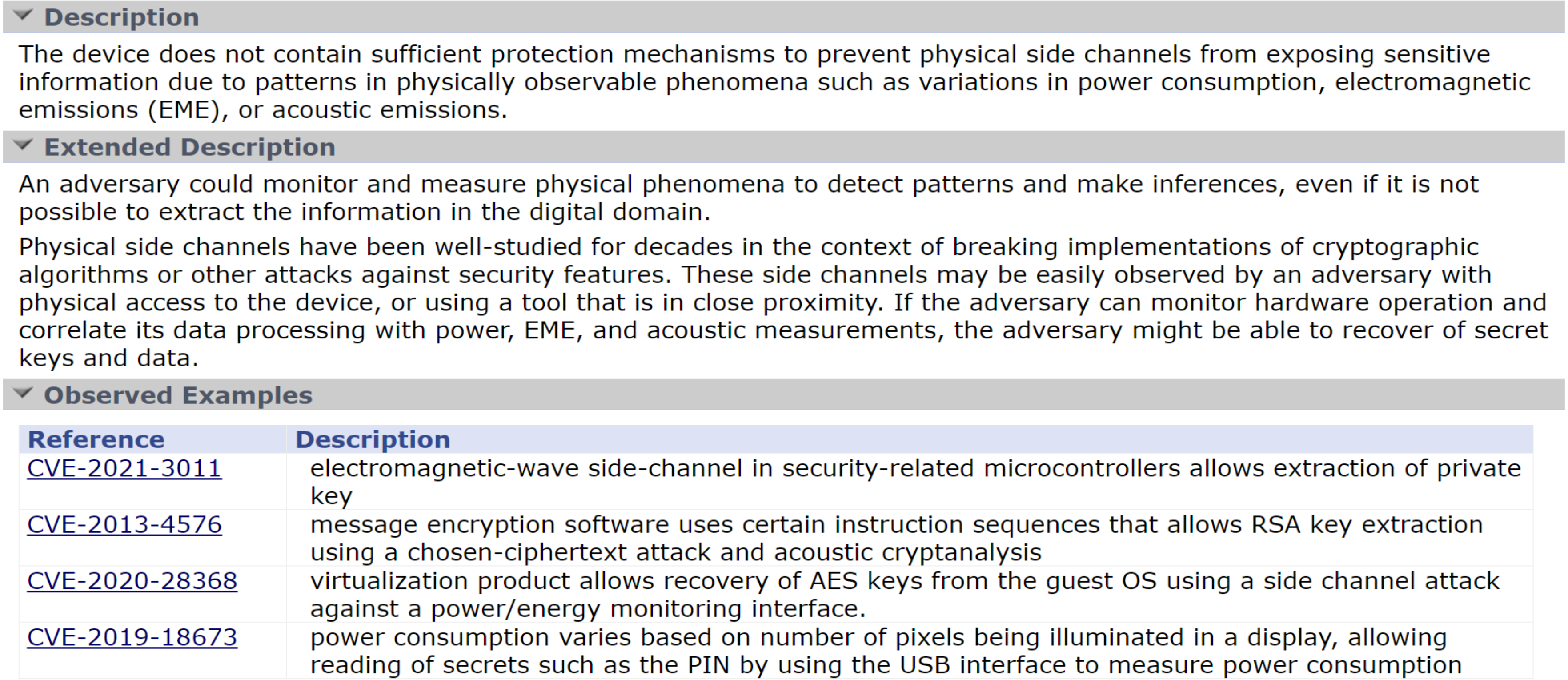
CWE Nit Bits

Bite-sized knowledge
that will enhance your CWE proficiency!

Observed Examples (OBEXs)

- Observed Examples are publicly reported vulnerabilities in real-world products that exhibit the weakness.
- This element is abbreviated as OBEX.
- Contains a brief description of the weakness seen in the example.
- Typically, will be a CVE reference, but doesn’t have to be.
- An entry, ideally, should have at least 1 OBEX.
- Contains a hyperlink to the source of the example.

CWE-1300: Improper Protection of Physical Side Channels



Description

The device does not contain sufficient protection mechanisms to prevent physical side channels from exposing sensitive information due to patterns in physically observable phenomena such as variations in power consumption, electromagnetic emissions (EME), or acoustic emissions.

Extended Description

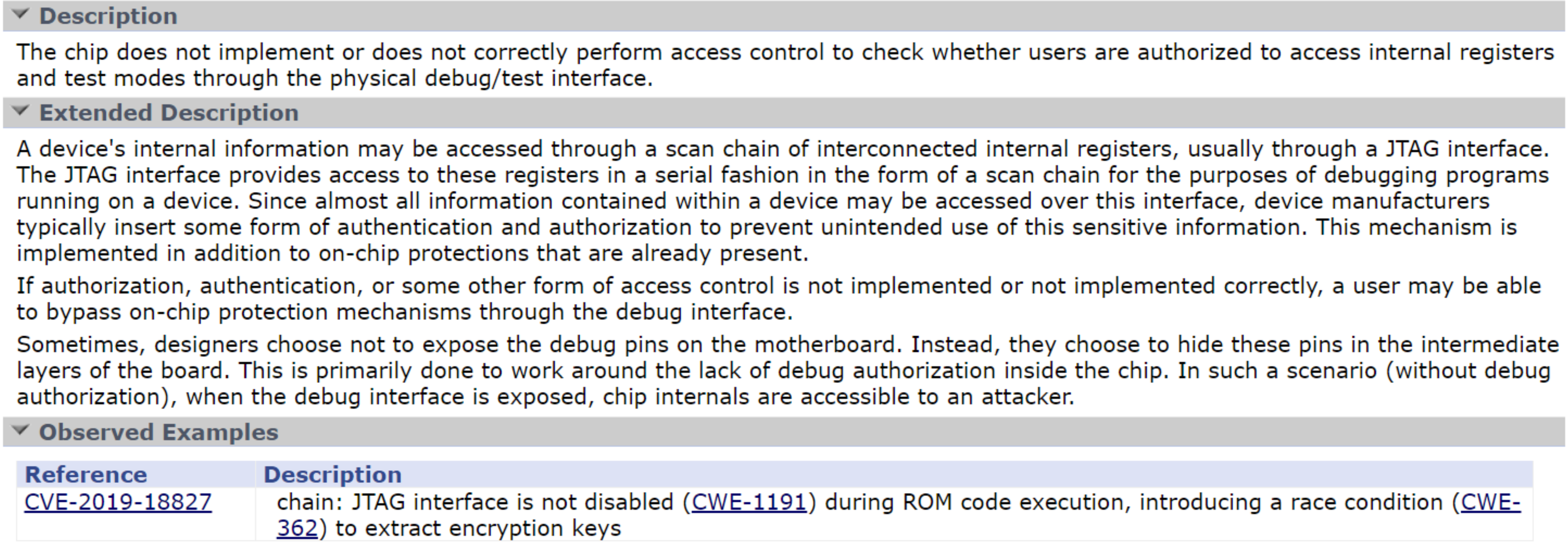
An adversary could monitor and measure physical phenomena to detect patterns and make inferences, even if it is not possible to extract the information in the digital domain.

Physical side channels have been well-studied for decades in the context of breaking implementations of cryptographic algorithms or other attacks against security features. These side channels may be easily observed by an adversary with physical access to the device, or using a tool that is in close proximity. If the adversary can monitor hardware operation and correlate its data processing with power, EME, and acoustic measurements, the adversary might be able to recover of secret keys and data.

Observed Examples

Reference	Description
CVE-2021-3011	electromagnetic-wave side-channel in security-related microcontrollers allows extraction of private key
CVE-2013-4576	message encryption software uses certain instruction sequences that allows RSA key extraction using a chosen-ciphertext attack and acoustic cryptanalysis
CVE-2020-28368	virtualization product allows recovery of AES keys from the guest OS using a side channel attack against a power/energy monitoring interface.
CVE-2019-18673	power consumption varies based on number of pixels being illuminated in a display, allowing reading of secrets such as the PIN by using the USB interface to measure power consumption

CWE-1191: On-Chip Debug and Test Interface With Improper Access Control



▼ Description

The chip does not implement or does not correctly perform access control to check whether users are authorized to access internal registers and test modes through the physical debug/test interface.

▼ Extended Description

A device's internal information may be accessed through a scan chain of interconnected internal registers, usually through a JTAG interface. The JTAG interface provides access to these registers in a serial fashion in the form of a scan chain for the purposes of debugging programs running on a device. Since almost all information contained within a device may be accessed over this interface, device manufacturers typically insert some form of authentication and authorization to prevent unintended use of this sensitive information. This mechanism is implemented in addition to on-chip protections that are already present.

If authorization, authentication, or some other form of access control is not implemented or not implemented correctly, a user may be able to bypass on-chip protection mechanisms through the debug interface.

Sometimes, designers choose not to expose the debug pins on the motherboard. Instead, they choose to hide these pins in the intermediate layers of the board. This is primarily done to work around the lack of debug authorization inside the chip. In such a scenario (without debug authorization), when the debug interface is exposed, chip internals are accessible to an attacker.

▼ Observed Examples

Reference	Description
CVE-2019-18827	chain: JTAG interface is not disabled (CWE-1191) during ROM code execution, introducing a race condition (CWE-362) to extract encryption keys

Incorrect Initializations

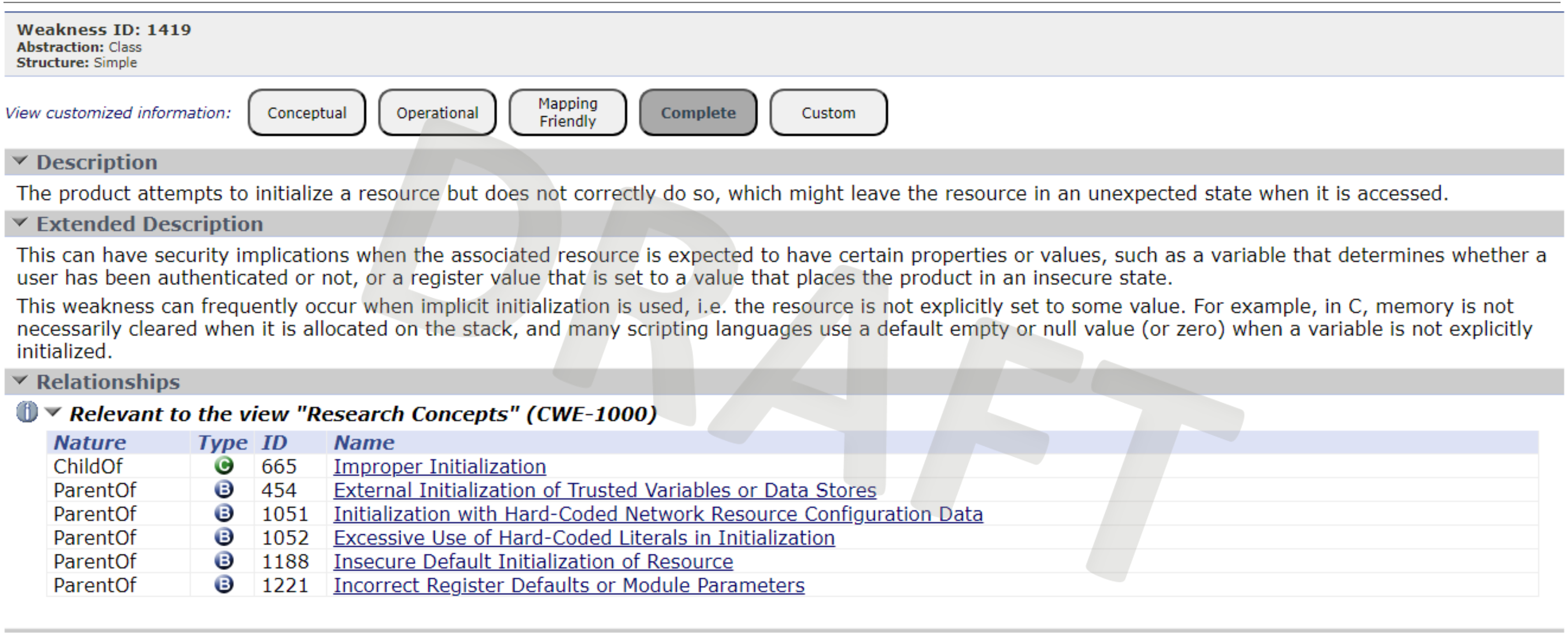
CWE Sneak Preview

Nordic APPROTECT Research

Research into the NORDIC APPROTECT (CVE-2020-27211) issue discussed at the last HW CWE SIG highlighted a gap within CWE.

- **(P) CWE-664: Improper Control of a Resource Through its Lifetime**
  - (C) CWE-665: Improper Initialization
    - (C) CWE-909: Missing Initialization of Resource
    - *(C) CWE-1419: Incorrect Initialization of Resource*
      - (B) CWE-1188: Insecure Default Initialization of Resource
      - (B) CWE-1221: Incorrect Register Defaults or Module Parameters
- **This new CWE-1419 entry will be in CWE 4.13 (October 26, 2023)**

CWE-1419: Incorrect Initialization of Resource



Weakness ID: 1419
Abstraction: Class
Structure: Simple

Description

The product attempts to initialize a resource but does not correctly do so, which might leave the resource in an unexpected state when it is accessed.

Extended Description

This can have security implications when the associated resource is expected to have certain properties or values, such as a variable that determines whether a user has been authenticated or not, or a register value that is set to a value that places the product in an insecure state.

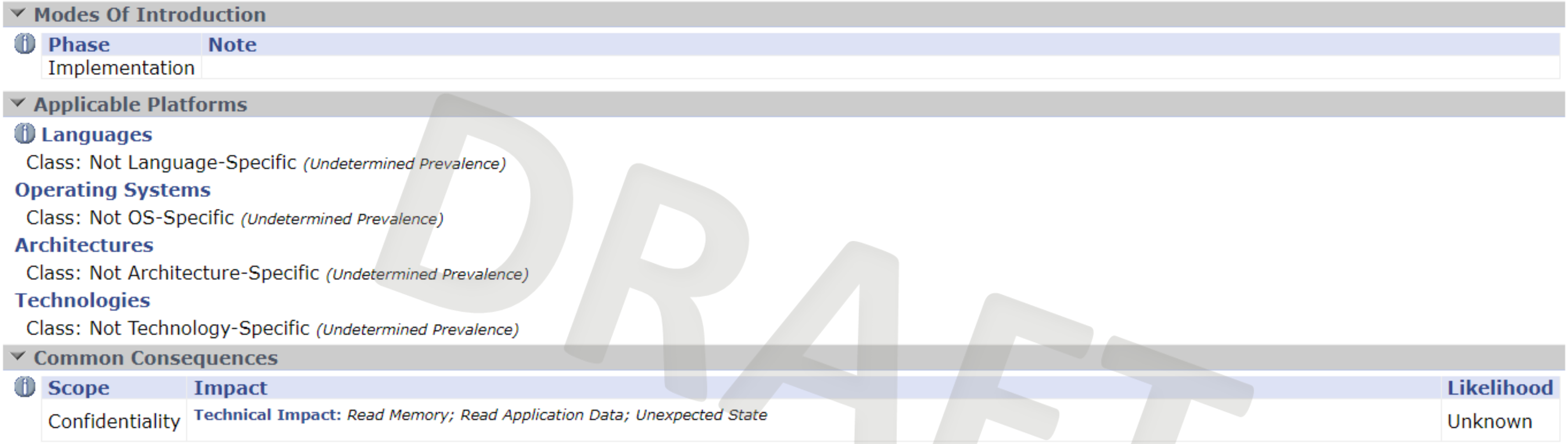
This weakness can frequently occur when implicit initialization is used, i.e. the resource is not explicitly set to some value. For example, in C, memory is not necessarily cleared when it is allocated on the stack, and many scripting languages use a default empty or null value (or zero) when a variable is not explicitly initialized.

Relationships

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

Nature	*Type*	*ID*	*Name*
ChildOf	C	665	Improper Initialization
ParentOf	B	454	External Initialization of Trusted Variables or Data Stores
ParentOf	B	1051	Initialization with Hard-Coded Network Resource Configuration Data
ParentOf	B	1052	Excessive Use of Hard-Coded Literals in Initialization
ParentOf	B	1188	Insecure Default Initialization of Resource
ParentOf	B	1221	Incorrect Register Defaults or Module Parameters

CWE-1419: Incorrect Initialization of Resource



Modes Of Introduction

Phase	Note
Implementation	

Applicable Platforms

Languages

Class: Not Language-Specific *(Undetermined Prevalence)*

Operating Systems

Class: Not OS-Specific *(Undetermined Prevalence)*

Architectures

Class: Not Architecture-Specific *(Undetermined Prevalence)*

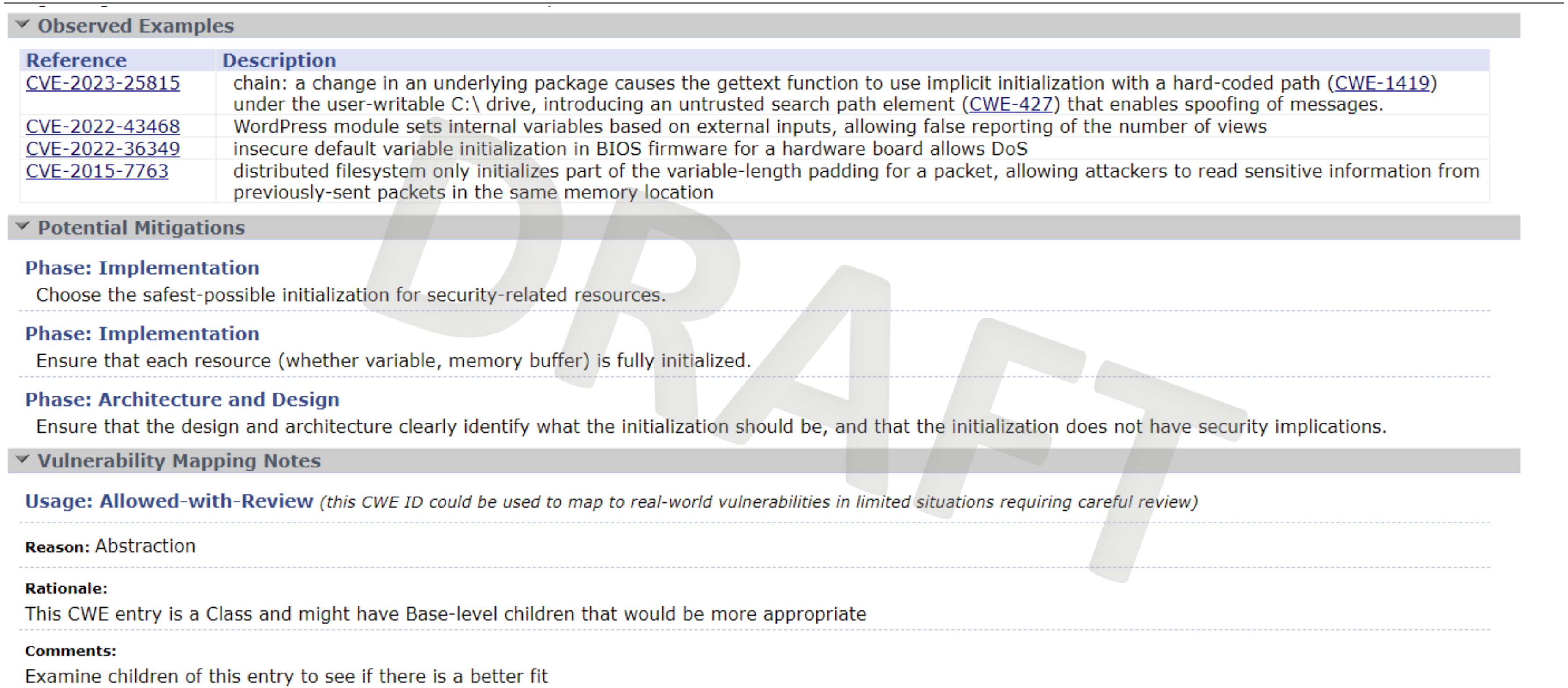
Technologies

Class: Not Technology-Specific *(Undetermined Prevalence)*

Common Consequences

Scope	Impact	Likelihood
Confidentiality	**Technical Impact:** *Read Memory; Read Application Data; Unexpected State*	Unknown

CWE-1419: Incorrect Initialization of Resource



Observed Examples

Reference	Description
CVE-2023-25815	chain: a change in an underlying package causes the gettext function to use implicit initialization with a hard-coded path (CWE-1419) under the user-writable C:\ drive, introducing an untrusted search path element (CWE-427) that enables spoofing of messages.
CVE-2022-43468	WordPress module sets internal variables based on external inputs, allowing false reporting of the number of views
CVE-2022-36349	insecure default variable initialization in BIOS firmware for a hardware board allows DoS
CVE-2015-7763	distributed filesystem only initializes part of the variable-length padding for a packet, allowing attackers to read sensitive information from previously-sent packets in the same memory location

Potential Mitigations

Phase: Implementation

Choose the safest-possible initialization for security-related resources.

Phase: Implementation

Ensure that each resource (whether variable, memory buffer) is fully initialized.

Phase: Architecture and Design

Ensure that the design and architecture clearly identify what the initialization should be, and that the initialization does not have security implications.

Vulnerability Mapping Notes

Usage: Allowed-with-Review *(this CWE ID could be used to map to real-world vulnerabilities in limited situations requiring careful review)*

Reason: Abstraction

Rationale:

This CWE entry is a Class and might have Base-level children that would be more appropriate

Comments:

Examine children of this entry to see if there is a better fit

Feedback Requested for CWE-1419 “Incorrect Initialization”

- **Should it be added to the hardware view?**
  - CWE-909 “Missing Initialization of Resource” is not in this view
    - One child is in the hardware view: CWE-1271: Uninitialized Value on Reset for Registers Holding Security Settings
    - Under category CWE-1206 Power, Clock, Thermal, and Reset Concerns
  - Maybe under “Cross-Cutting Problems” (1208)?

CWE and CAPEC are sponsored by U.S. Department of Homeland Security (DHS) Cybersecurity and Infrastructure Security Agency (CISA).

Feedback Requested: Elements that Could be Enhanced in CWE-1419

- **See screenshots in previous slides**
- **Modes of Introduction**
  - Phases: Implementation, System Configuration, Operation, Installation, Manufacturing
- **Potential mitigations**
  - Minimal explanation; are stronger options available?
- **Common consequences**
  - Currently limited to confidentiality
- **Observed examples**
  - CVE-2022-36349 (BIOS for a hardware board)
- **Demonstrative examples**
  - One example reused from child CWE-1221: Incorrect Register Defaults or Module Parameters

Inclusive Language and HW CWEs (Inclusive Terminology)

Inclusive Language in CWE

- **2006: gender-neutral pronouns since CWE's creation**
- **2020/2021: Focus on other areas until at least CWE 4.5**
  - "Whitelist" (allowlist)/"blacklist" (denylist)
  - "man-in-the-middle" → "adversary-in-the-middle"
  - "Master"/"Slave" was rarely used
- **2020 to today: significant increase in external submissions (e.g., hardware for CWE 4.0)**
- **2020 to today: software/standards community efforts**
  - IETF draft "Terminology, Power, and Inclusive Language in Internet-Drafts and RFCs"
  - NISTIR 8366 - Guidance for NIST Staff on Using Inclusive Language in Documentary Standards
  - ACM: "Words Matter - Alternatives for Charged Terminology in the Computing Profession"

Inclusive Language in Semiconductor Industry

- Arm’s “Inclusive Language Commitment: Arm is committed to making the language we use inclusive, meaningful, and respectful. Our goal is to remove and replace non-inclusive language from our vocabulary to reflect our values and represent our global ecosystem.”[1]
- Xilinx has their Inclusive Naming Initiative[2]
- Numerous examples of Intel updating documentation to use inclusive terminology[3]

1. https://www.arm.com/en/company/sustainability/business-practices
2. https://www.amd.com/content/dam/amd/en/documents/corporate/cr/Inclusive-terminology.pdf
3. https://www.intel.com/content/www/us/en/docs/programmable/683609/21-3/creating-a-system-with-revision-history.html

How to Assist in Inclusive Language for HW CWEs?

- **Some HW CWE examples with non-inclusive language:**
  - CWE-1318: “bus master… from slave to master” etc.
  - CWE-1267 and CWE-1290: Demox uses “bus master”
  - CWE-1317: Ext desc uses master/slave.
  - Other CWEs: 1193, 1248, 1264, 1274, 1290, 1315, 1331
- **Submitted demox’s were able to use alternate terminology.**
- **Please email cwe@mitre.org any standards or references for alternate terminology for hardware.**

Open *Community Items*

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

- **Missing Mitigations**
  - 4 CWEs are missing mitigations (No change)
- **Missing demonstrative examples (DEMOX)**
  - 16 CWEs missing demonstrative examples (down 1)
    - 1 added from Hack@DAC, CWE-325
    - Note: there are other DEMOXs from Hack@DAC but now adding DEMOXs to entries that have an existing DEMOX
- **Missing Observed Examples (OBEX)**
  - 66 CWEs do not have any observed examples (down 1)

Thank you contributors. Keep them coming.

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

Resonant Frequency Weakness

- **If anyone is interested is developing and proposing a resonant frequency weakness, please see discussion points on GitHub.**
- **https://github.com/CWE-CAPEC/hw-cwe-sig/issues/105**

Next Meeting (Nov 10th or Nov 3rd)

CWE@MITRE.ORG

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

Backup