Hardware CWE™ Special Interest Group (SIG)

Chair: Bob Heinemann (MITRE)

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

MITRE Team: Gage Hackford, Steve Christey Coley, Alec Summers

MITRE

September 13, 2024

Agenda

REMINDER: This meeting is being recorded.

1	System Verilog and Schema	Bob	5 Min
2	The Hack@DAC Story	Arun Kanuparthi (Intel)	20 min
3	Covert Channel Recommendations	Bob / Manna	20 min
4			

Housekeeping

- **Schedule:**
  - **Next Meeting:Oct 11**
    - **12:30 – 1:30 PM EST (16:30 – 17:30 UTC)**
    - **Microsoft Teams**
- **Contact: cwe@mitre.org**
- **Mailing List:** *hw-cwe-special-interest-group-sig-list@mitre.org*
- **Minutes from previous meetings available on our GitHub site:**
  - **https://github.com/CWE-CAPEC/hw-cwe-sig**

Announcements

- **CWE Content Development Repository (CDR) pilot now on GitHub! Open to anyone by request. Public access in the next few months.**
- **CWE 4.16 release is planned for October.**
- **CWE 5.0 is planned for early 2025.**

Call for Topics

What topics should we cover next time?

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

System Verilog

System Verilog and Verilog

- After discussion last meeting the decision was made to:
  - Add System Verilog to the schema
  - Change everything to be SystemVerilog since it is a superset of Verilog.
  - If there are **no objections** we'll proceed to implement this into the next release.

HACK@DAC Presentation

The Hack@DAC Story: Black Hat USA/Asia 24

- Full version of slides can be obtained here

The Hack@DAC* Story:
Learnings from Organizing the World's Largest Hardware Hacking Competition

Arun Kanuparthi, Hareesh Khattri, Jason Fung (Intel Corporation, USA)

JV Rajendran (Texas A&M University, USA), Ahmad-Reza Sadeghi (TU Darmstadt, Germany)

*Design Automation Conference

The Team

Arun Kanuparthi
Principal Engineer,
Offensive Security Researcher
Intel Corporation, USA

Hareesh Khattri
Principal Engineer,
Offensive Security Researcher
Intel Corporation, USA

Jason Fung
Sr. Director
Offensive Security Research
Intel Corporation, USA

Jeyavijayan (JV) Rajendran
Associate Professor
Texas A&M University, USA

Ahmad-Reza Sadeghi
Professor
TU Darmstadt, Germany

Offensive Security Research at Intel

- 50+ years of combined experience
- CPUs, Servers, Clients, Networking, Cellular, Storage, Security technologies, …
- 500+ vulnerabilities identified
- Vulnerability root causing and categorization
- MITRE HW CWE SIG* members

Security Research

- 35+ years of combined experience
- Circuits, system security, network security, cryptography, microarchitecture, etc.
- 44000+ citations!

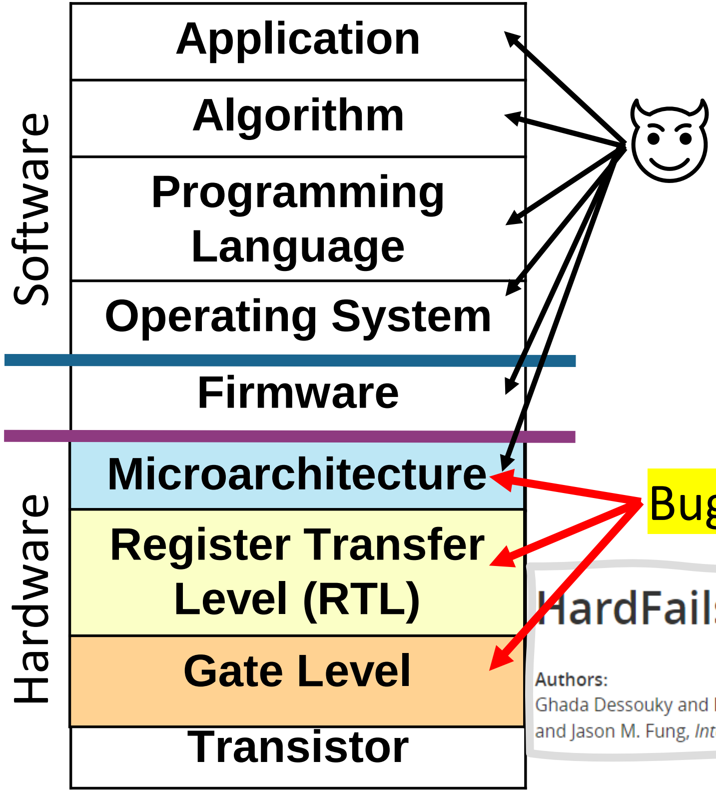
*Special Interest Group (SIG)

Overview

- Introduction
- Value of Organizing HW CTFs
- How Hack@DAC is Unique
- Organizing Hack@DAC
- Key Takeaways & Summary

Race to the Bottom of the Stack

Challenge #1: Limited Awareness of HW Security Weaknesses



Bugs in hardware could be exploitable by software!

HardFails: Insights into Software-Exploitable Hardware Bugs

Authors:
Ghada Dessouky and David Gens, *Technische Universität Darmstadt;* Patrick Haney and Garrett Persyn, *Texas A&M University;* Arun Kanuparthi, Hareesh Khattri, and Jason M. Fung, *Intel Corporation;* Ahmad-Reza Sadeghi, *Technische Universität Darmstadt;* Jeyavijayan Rajendran, *Texas A&M University*

USENIX Security 2019

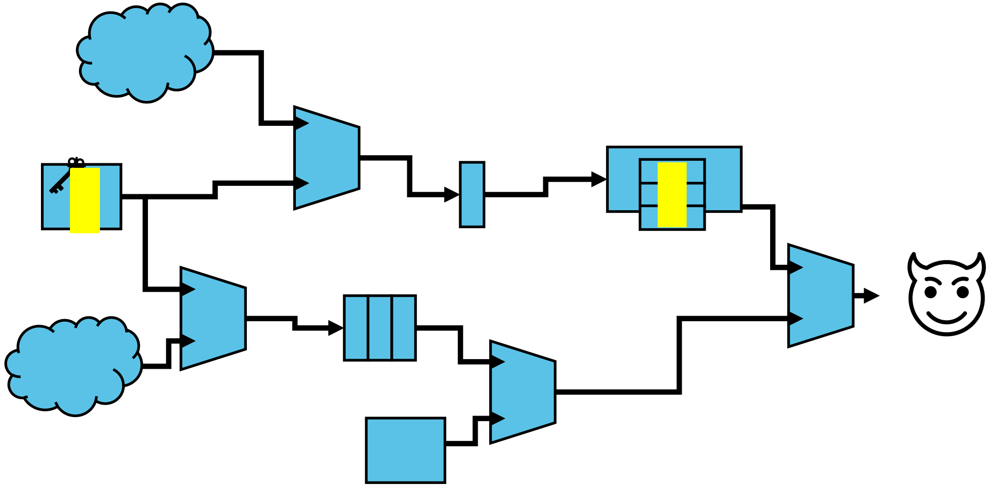
Tools for Security – SW vs HW

Challenge #2: Need for Security-Aware Design Automation Tools

Software
- Application
- Algorithm
- Programming Language
- Operating System

- Firmware

Hardware
- Microarchitecture
- Register Transfer Level (RTL)
- Gate Level
- Transistor

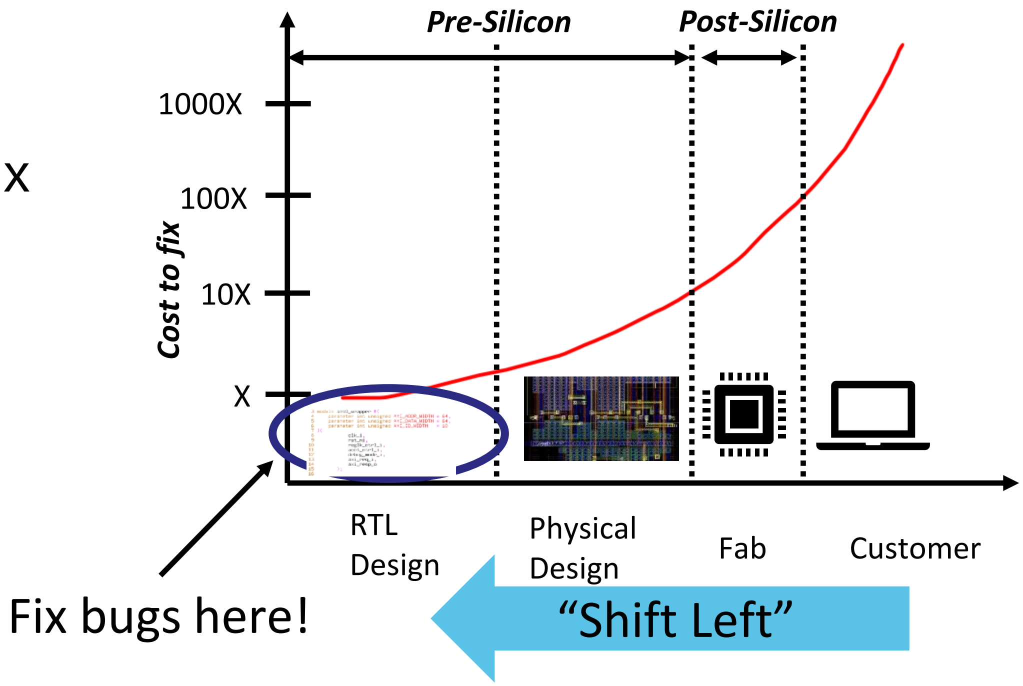


HW security tools (at RTL level) are limited

Cost of Fixing Bugs

Challenge #3: Need to Detect/Fix Bugs at RTL Design Phase

- SW bugs fixed with patches
- HW bugs are complicated to fix
  - Time consuming
  - Expensive
  - Cause brand damage



Motivation for Hack@DAC

Awareness of Hardware Common Weaknesses

CONCEPTS

Security-Aware Design Automation

TOOLS

"Shift-Left" to Detect & Fix Bugs in RTL

BEST PRACTICES

Hack@DAC

- Hackathons, trainings
- Open-source hardware as target?
- What about hardware CTF?

Overview

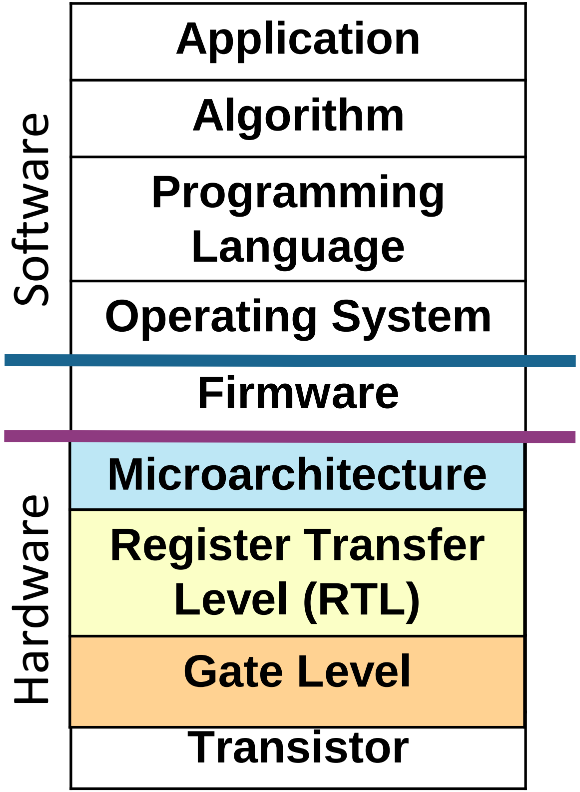
Fostering Awareness for HW Security

- Continuous race between attackers and defenders
- Defenders need to up their game!
- Hardware CTFs foster greater awareness about
  - Common hardware security weaknesses
  - Constraints of chip design teams

Image source

Overview

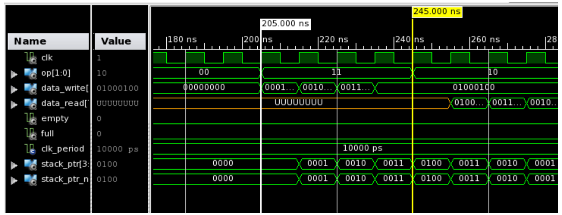
Popular HW CTFs



- Popular HW CTFs are “closed-box”
- Adopt a hacker-centric approach
  - Involve physical interaction with target chip
    - Probing input/output ports
    - Desoldering and reverse engineering attacks
    - Physical side channel attacks, etc.
  - No insights into the RTL code of the chip
- Very important research!
- Does not address “shift-left” challenge

Closed-box vs Open-box CTFs

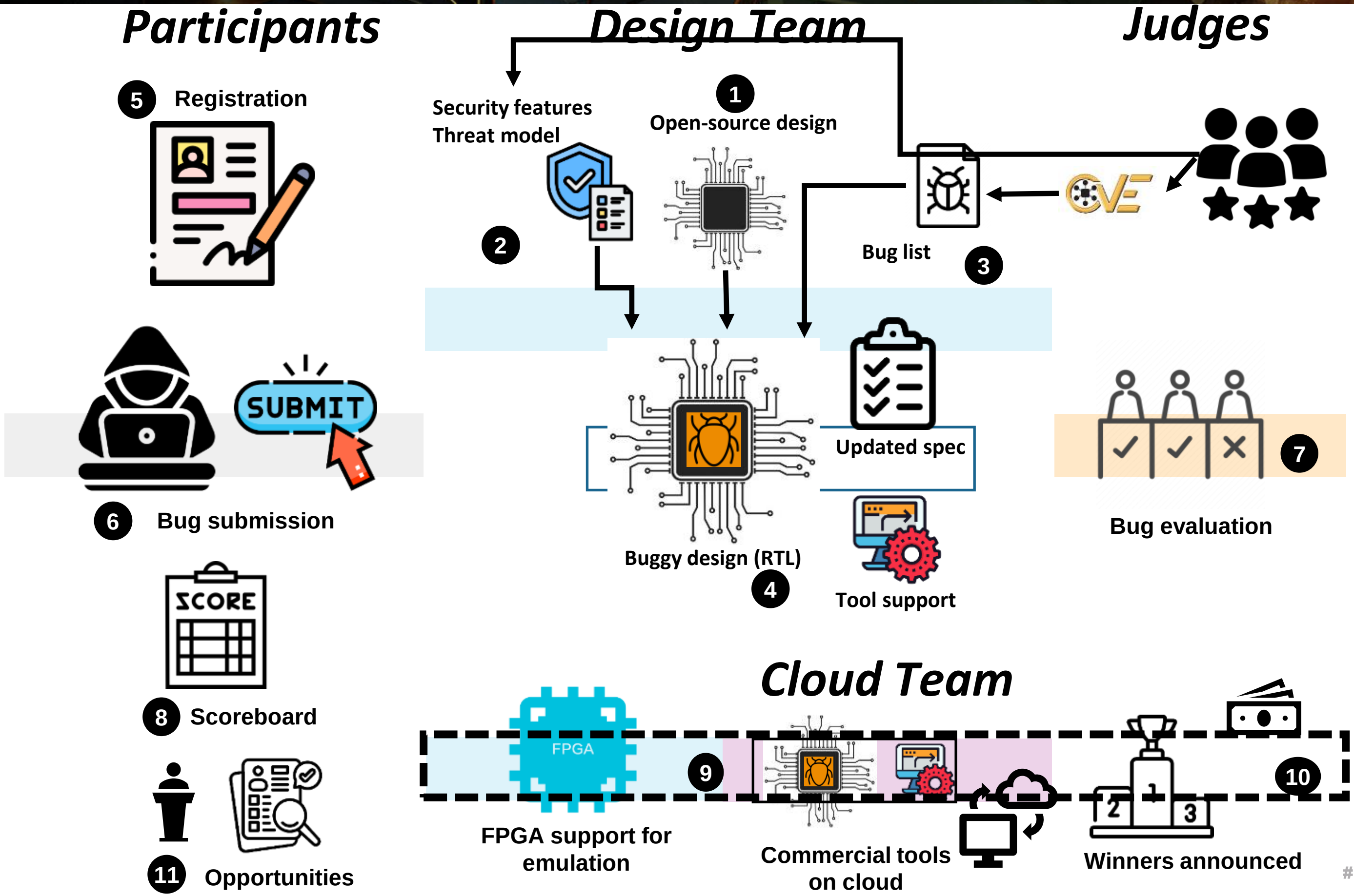
- Hack@DAC is “Open-box”
  - Participants given a buggy SoC RTL
  - Finer grained scope
- Participants attempt to break security features
  - RTL Simulation/ Emulation
  - Formal Verification
  - RTL Static Analysis
  - Manual reviews
- **Designer-centric approach**



Overview

- Introduction
- Value of Organizing HW CTFs
- How Hack@DAC is Unique
- **Organizing Hack@DAC**
- Key Takeaways & Summary

Hack@DAC – The Process



Competition: Phase 1

- Phase 1 is offline
- Participants have over 2 months to:
  - Analyze entry points
  - Identify assets
  - Develop security test cases
  - Develop custom tools to detect bugs
  - Submit bugs for evaluation by judges
- Extended duration allows for equal access to participants from various backgrounds.

Submission and Scoring

B	D	E	F	G	H	I	J	K	L	N
Team name	Security feature bypassed	Finding	Location or code reference	Detection method	Security impact	Adversary profile	Proposed mitigation	CVSSv3.1 score and severity	CVSSv3.1 Details	Judges comments
					other wrappers, all the secure data can be read out.					
	Register Lock Control signal unset	In access control register wrapper file, reglk_ctrl signal is responsible for reading/writing the signal for locked peripherals. All bits set to '1' of reglk_ctrl signal indicates the peripheral is locked otherwise bits set to '0' indicate normal operation. Therefore, by default reglk_ctrl should always be set high to prevent unauthorized access. We found that only lower half of the reglk_ctrl is set from 8-bit input reglk_ctrl_i and higher bits are set to 0. Thus, all bits from 8-15 are set to 0 and should not be accessed for any read/write operation. In acc_wrapper.sv, at line 96, 98 and	piton/design/chip/tile/ariane/src/acct/acct_wrapper.sv, Line 96, 98 and 100.	Manual analysis + User level assertion generation + Formal property verification using Synopsys VCStatic	This bug will lead to accessing peripheral device even when its register is in locked state (which ideally should have restricted its access).	Unprivileged software at user-level mode	One line verilog change in acct_wrapper.sv: reglk_ctrl[13] -> reglk_ctrl[3]	Medium (6.1)	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:L/I:H/A:N/RC:C Attack vector: Local. A person having read/write/execute access on the SoC can mount the attack. Attack complexity: Low. An exploit code developed can sureshot obtain access control of	Valid issue (5) + correc impact analysis (5) + FPV usage to detect bug (50)

Competition: Phase 2 (Finals)

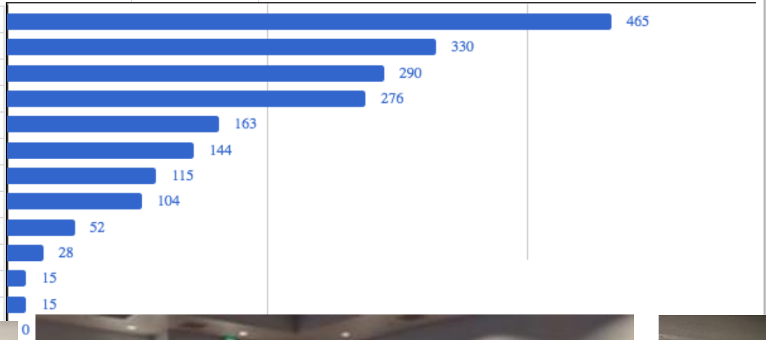
- Top 10 teams invited to participate in finals
- Phase 2 live at the conference
- Partnership with Synopsys
  - All necessary tools hosted on Synopsys cloud
  - Buggy design ported to cloud
  - Tool trainings provided to all finalists
- Travel grants to US-based finalists to attend in person
- Duration of 48 hours

Competition: Phase 2 (Finals)

Live Scoreboard

Hack@DAC'19 Beta Scoreboard : Live

Team name	Points
Hackin' Aggies*	465
NOPS	330
Always@Posedge	290
NotATrojan	276
Alpha4	163
.:hackamole:.	144
SEC	115
Team 11	104
Chipsters	52
Tribe	28
CCNY	15
CICA*	15



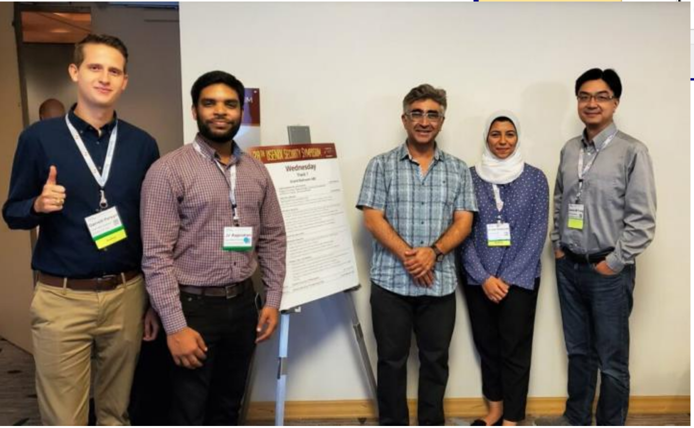


Image: “Hacking SoC IP Under Pressure”, SemiEngineering 2018 source

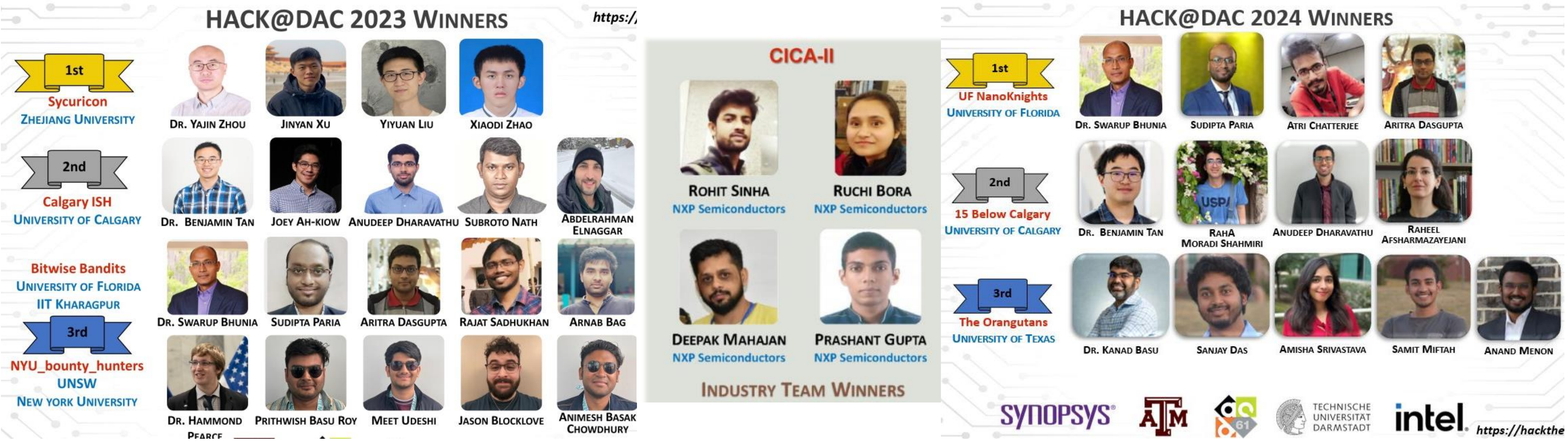
Competition: Phase 2 (Finals)

Winners Honored

Publications

So Far..

- Extended to USENIX Security (Hack@SEC) and CHES (Hack@CHES)
- 300+ teams participated from all over the world; 1000+ participants
- Industry participation too!
- Past winners now working in hardware security roles at top companies



Overview

- Introduction
- Value of Organizing HW CTFs
- How Hack@DAC is Unique
- Organizing Hack@DAC
- Key Takeaways & Summary

Recap of 3 Top Challenges

Awareness of Hardware Common Weaknesses

Security-Aware Design Automation

“Shift-Left” to Detect & Fix Bugs in RTL

Awareness of HW Weaknesses

MITRE Hardware CWE

https://cwe.mitre.org

1194 - Hardware Design
- 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)*

- 75+/110 CWE entries contributed by Intel
- Hack@DAC vulnerability and mitigation examples now added to several CWE entries
- "Hardware Security Failure Scenarios"

CWE-1245: Improper Finite State Machines (FSMs) in Hardware Logic

Weakness ID: 1245
Vulnerability Mapping: ALLOWED
Abstraction: Base

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

Description

Faulty finite state machines (FSMs) in the hardware logic allow an attacker to put the system in an undefined state, to cause a denial of service (DoS) or gain privileges on the victim's system.

Extended Description

The functionality and security of the system heavily depend on the implementation of FSMs. FSMs can be used to indicate the current security state of the system. Lots of secure data operations and data transfers rely on the state reported by the FSM. Faulty FSM designs that do not account for all states, either through undefined states (left as don't cares) or through incorrect implementation, might lead an attacker to drive the system into an unstable state from which the system cannot recover without a reset, thus causing a DoS. Depending on what the FSM is used for, an attacker might also gain additional privileges to launch further attacks and compromise the security guarantees.

Relationships

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

Nature	Type	ID	Name
ChildOf		684	Incorrect Provision of Specified Functionality

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

Nature	Type	ID	Name
MemberOf	C	1199	General Circuit and Logic Design Concerns

Modes Of Introduction

Phase	Note
Architecture and Design	
Implementation	

Applicable Platforms

Languages

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

Operating Systems

Security-Aware Tooling & Bug Detection

- Security Test Case Generation and Bug Patching using GenAI/ LLMs
  - (Security) Assertions by Large Language Models *(IEEE TIFS 2024)*
  - Examining Zero Shot Vulnerability Repair with Large Language Models *(IEEE Security and Privacy*
  - Fixing Hardware Security Bugs with Large Language Models *(arXiv)*
  - On Prompting Hardware Security Bug Code Fixes by Prompting Large Language Models *(IEEE TIF*
  - DIVAS: An LLM-based End to End Framework for SoC Security Analysis and Policy-based Protection *(arXiv)*
- Formal Verification
  - Sylvia: Countering the Path Explosion Problem in the Symbolic Execution of Hardware Designs *(FMC*
  - All Artificial, Less Intelligence: GenAI Through the Lens of Formal Verification *(arXiv)*
- Static Analysis
  - Don't CWEAT It: Toward CWE Analysis Techniques in Early Stages of Hardware Design *(IEEE/ACM I*
- Concolic Testing
  - RTL-ConTest: Concolic Testing on RTL for Detecting Security Vulnerabilities *(IEEE TCAD 2022)*
- Hardware Information Flow Tracking

Key Takeaways for Academia

- Hack@DAC SoC framework
  - Realistic threat model and security objectives
  - Closest available to commercial chip designs
  - Uncover new classes of security vulnerabilities
- Get invaluable hardware security assurance skills!
  - Mimic security teams at a chip design company
  - Develop a hacker mindset
- Competition format
  - provides equal access to participants from diverse backgrounds
    - Strong technical female participation
  - Facilitates participation from various geos/ time zones

Hack@DAC 2018 finals at San Francisco, CA

Image: "Hacking SoC IP Under Pressure", SemiEngineering 2018 source

Takeaways for Industry

- Improve in-house security assurance best practices
  - Exposure to new kinds of weaknesses
  - Planning for survivability features
  - Easier for functional verification teams to pick up security assurance
- New tools for identifying weakness classes
  - Publish guides on detection of classes of hardware security weaknesses
- Add security capabilities to today's functional tools
  - Address gaps of today's security verification tools to detect classes of vulnerabilities

Media Coverage

EE Times

Capture-the-Flag Competitions Need to Include Hardware

Cyber Defense Magazine

Learning Hardware Security Via Capture-The-Flag Competitions

DevOps digest

Why Do We Need a Standardized Framework to Enumerate Hardware Security Weaknesses?

techspective

Intel Hardware CTF Competitions Drive Innovation for Next-Gen Secure Computing Platforms

Semiconductor Engineering

Hacking SoC IP Under Pressure

Dark Reading

Intel Harnesses Hackathons to Tackle Hardware Vulnerabilities

USA 2024

Covert Channel Recommendation for HW (Work in Progress)

Context

- **The following is a set of ideas, observations, and recommendations for discussion and nothing is finalized.**
- **The intent of the following slides is to present recommendations on how to improve Covert Channel coverage for HW CWE based on community feedback.**
- **For us to move forward with any changes we would like to get buy-in from the community and solicit support in implementation.**
- **Any proposed changes are also pending review from the CWE tech lead.**

Covert Channels Discussion Summary

Member Comments

- Covert Channels should have coverage in the hardware view *–Jason Oberg*
- Covert Channels should be in the HW categories Security Flow Issues, General Circuit and Logic Design Concerns, or Debug and Test Problems. *–Paul Wortman*
- CWE-514 as currently written it's specific to software and would need to be tweaked *–Bruce Monroe*
- **May / June HW SIG Meeting**
  – Bob / Manna presented current CWE coverage on covert channels as well as the concept of incidental channels
- **July HW SIG Meeting**
  – Hareesh discussed the importance of considering designers intent when considering covert channels. Suggested that this should be considered for the relationships to the CWE covert channel entry CWE-514.

Incidental Channels concept and relation to CWE-1229: Creation of Emergent Resource

Intel's concept of "Incidental channels":

- In computing systems Incidental Channels are unintended communication channels formed by valid properties such as execution time, power consumption, and the use of shared resources. When data flows through an incidental channel, both data values and metadata (for example, memory addresses being accessed) may be inferable by malicious actors.

Description

The product manages resources or behaves in a way that indirectly creates a new, distinct resource that can be used by attackers in violation of the intended policy.

Extended Description

A product is only expected to behave in a way that was specifically intended by the developer. Resource allocation and management is expected to be performed explicitly by the associated code. However, in systems with complex behavior, the product might indirectly produce new kinds of resources that were never intended in the original design. For example, a covert channel is a resource that was never explicitly intended by the developer, but it is useful to attackers. "Parasitic computing," while not necessarily malicious in nature, effectively tricks a product into performing unintended computations on behalf of another party.

Recommend to create new CWE based on incidental channels and organize under CWE-1229

Covert Channels

- From Intel's treatment on Incidental Channels, "The threat model of covert channels requires attackers to be able to access relevant, secret information before exposing it via the covert channel."

- This means that an incidental channel as a weakness does not lead to the introduction of a vulnerability unless there is another weakness present that allows unauthorized access to data.

- CWE has a way to model that type of relationship and is referred to as composites.

Recommend we explore how to represent the current Covert Channel Weakness (CWE-514) as a composite of Incidental Channel and Improper Access Control.

Adding to HW View

These were the categories that have been suggested (Paul Wortman):

Security Flow Issues: related to improper design of full-system security flows, including but not limited to secure boot, secure update, and hardware-device attestation.

General Circuit and Logic Design Concerns: 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.

Debug and Test Problems: related to hardware debug and test interfaces such as JTAG and scan chain.

Also consider the following since a key issue with covert channel is access to data:

Privilege Separation and Access Control Issues: 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.

Recommend we put new incidental channel CWE and Covert Channel CWE in the Security Flow Issue and Privilege Separation and Access Control Issues Categories in HW View.

Designer’s Intent

Should these (Covert Channel Weaknesses) be updated to clarify and emphasize that design intent part or is just observable discrepancy sufficient for a weakness?

- A designer can make specific claims of what their product protects and not protects against or they can make no claims at all.
- The definition of a weakness does not take designer’s intent into account.
- A condition in a software, firmware, hardware, or service component that, under certain circumstances, could contribute to the introduction of vulnerabilities.
- If we feel this is an important point to make when discussing covert channels we can express it somewhere other than the description.

Updating CWE-514 to be less software centric

The description and extended description appear to be generic enough to cover both hardware and software.

The demonstrative example could have a HW example.

The current DEMOX appears to be an example of a side channel. This will need further review.

Recommend updating entry to include a HW focused DEMOX and revisit inclusion of current DEMOX.

Recommendations for discussion

- **Key Item: Establish composite relationship between Incidental Channels and Access Control for Covert Channels**
- **Update CWE-1229 (Emergent Resource) to be less software centric.**
- **Create new CWE "Creation of Incidental Channel" Class (Child of 1229)**
- **Put this new CWE and CWE-1229 into the HW View (which category?)**
- **Organize Covert Channel under the new CWE, put that into the HW view.**
- **The Covert Channel CWE should also specify a composite relationship**
- **Update covert channel entry to have hw specific DEMOX and OBEX?**

Next Meeting (Oct 11)

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.**