Cycuity

Using Metrics to Drive a Proactive Hardware Security Program

Jason Oberg (jason@cycuity.com)

Co-founder and CTO

About Me

- **Co-founder Chief Technology Officer of Cycuity**
    - Cycuity provides software products to increase security assurance in semiconductor designs
- **PhD in Hardware Security from UC San Diego**
- **15+ years in semiconductors and security**
- **Member of the Common Weakness Enumeration (CWE) board**

Cycuity

Using Metrics to Drive a Security Development Lifecycle

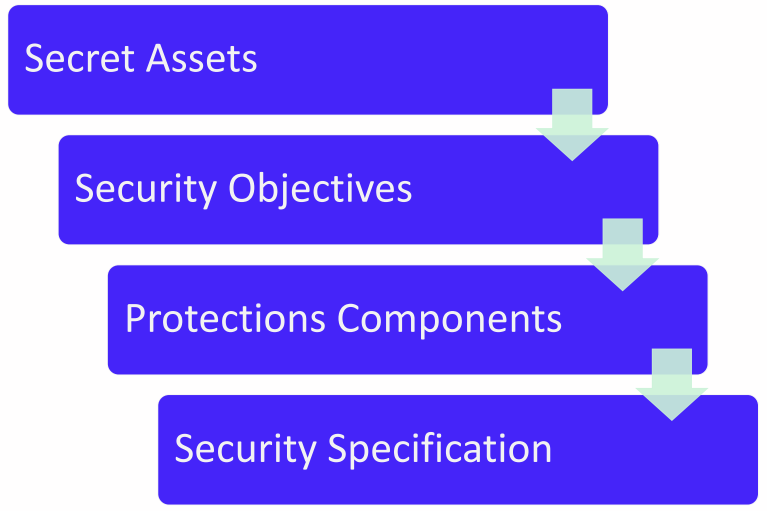
Security Requirements
Collected before or during early stage of system development

Security Verification
Regularly applied as chip and system is developed

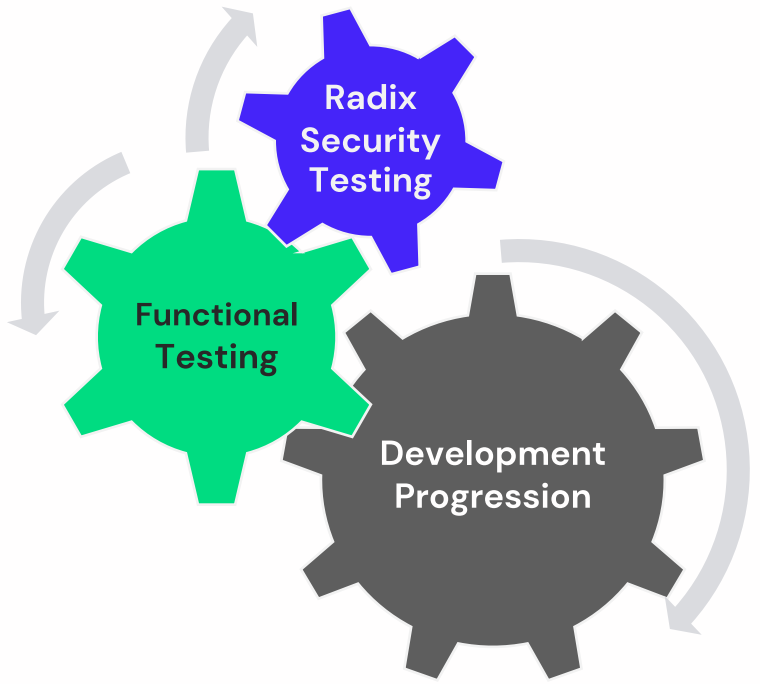
Security Signoff
Confirmed before manufacturing

Cycuity helps customers to…

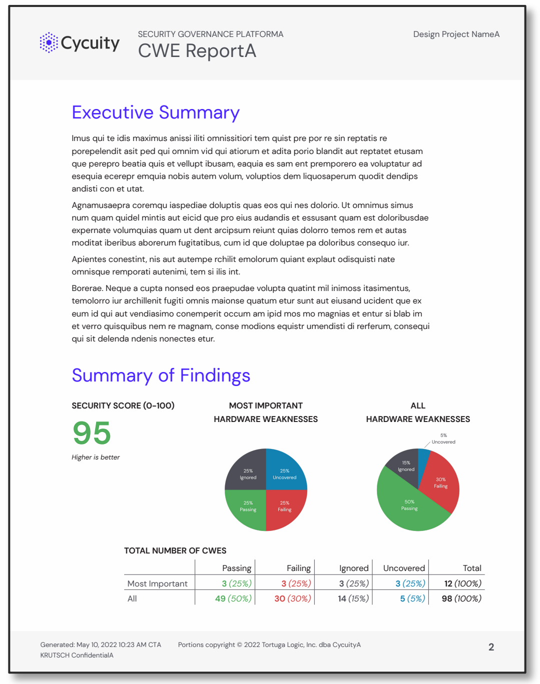
Align stakeholders on business needs and **verifiable security requirements**



Apply **automated security verification** at each stage of the design process



Document **security compliance** by providing verification evidence



Cycuity SECURITY GOVERNANCE PLATFORMA
CWE ReportA
Design Project NameA

Executive Summary

Imus qui te idis maximus anissi iliti omnissitiori tem quist pre por re sin reptatis re porepelendit asit ped qui omnim vid qui atiorum et adita porio blandit aut reptatet etusam que perepro beatia quis et vellupt ibusam, eaquia es sam ent premporero ea voluptatur ad esequia ecerepr emquia nobis autem volum, voluptios dem liquosaperum quodit dendips andisti con et utat.

Agnamusaepra coremqu iaspediae doluptis quas eos qui nes dolorio. Ut omnimus simus num quam quidel mintis aut eicid que pro eius audandis et essusant quam est doloribusdae expernate volumquias quam ut dent arcipsum reiunt quias dolorro temos rem et autas moditat iberibus aborerum fugitatibus, cum id que doluptae pa doloribus consequo iur.

Apientes conestint, nis aut autempe rchilit emolorum quiant explaut odisquisti nate omnisque remporati autenimi, tem si ilis int.

Borerae. Neque a cupta nonsed eos praepudae volupta quatint mil inimoss itasimentus, temolorro iur archillenit fugiti omnis maionse quatum etur sunt aut eiusand ucident que ex eum id qui aut vendiasimo conemperit occum am ipid mos mo magnias et entur si blab im et verro quisquibus nem re magnam, conse modions equistr umendisti di rerferum, consequi qui sit delenda ndenis nonectes etur.

Summary of Findings

SECURITY SCORE (0–100)
95
Higher is better

MOST IMPORTANT HARDWARE WEAKNESSES
25% Ignored
25% Uncovered
25% Passing
25% Failing

ALL HARDWARE WEAKNESSES
5% Uncovered
15% Ignored
30% Failing
50% Passing

TOTAL NUMBER OF CWES

	Passing	Failing	Ignored	Uncovered	Total
Most Important	3 *(25%)*	3 *(25%)*	3 *(25%)*	3 *(25%)*	12 *(100%)*
All	49 *(50%)*	30 *(30%)*	14 *(15%)*	5 *(5%)*	98 *(100%)*

Generated: May 10, 2022 10:23 AM CTA
KRUTSCH ConfidentialA
Portions copyright © 2022 Tortuga Logic, Inc. dba CycuityA
2

What is CWE?

Common root-cause *weaknesses* that can lead to *vulnerabilities*

1194 - Hardware Design View

https://cwe.mitre.org/data/definitions/1194.html

⊞ C Manufacturing and Life Cycle Management Concerns - *(1195)*

- Defects that arise in the semiconductor-manufacturing process.

⊞ C Security Flow Issues - *(1196)*

- Full-system security flows, including secure boot, secure update, and hardware-device attestation.

⊞ C Integration Issues - *(1197)*

- Integration of multiple IP cores, from SoC subsystem interactions, or from hardware platform subsystem interactions.

⊞ C Privilege Separation and Access Control Issues - *(1198)*

- Hardware-based isolation and access control (e.g., identity, policy, locking control) of sensitive shared hardware resources such as registers and fuses.

⊞ C General Circuit and Logic Design Concerns - *(1199)*

- Hardware-circuit design and logic as well as issues related to hardware description languages such as System Verilog and VHDL.

⊞ C Core and Compute Issues - *(1201)*

- CPUs, Graphics, Vision, AI, FPGA, and microcontrollers.

⊞ C Memory and Storage Issues - *(1202)*

- Memory (e.g., DRAM, SRAM) and storage technologies (e.g., NAND Flash, OTP, EEPROM, and eMMC).

⊞ C Peripherals, On-chip Fabric, and Interface/IO Problems - *(1203)*

- Peripheral devices, I/O interfaces, on-chip interconnects, NoCs, and buses.

⊞ C Security Primitives and Cryptography Issues - *(1205)*

- Cryptographic protocols and other hardware-security primitives such as PUFs and RNGs.

⊞ C Power, Clock, and Reset Concerns - *(1206)*

- System power, voltage, current, temperature, clocks, system state saving/restoring, and resets at the platform and SoC level.

⊞ C Debug and Test Problems - *(1207)*

- Hardware debug and test interfaces such as JTAG and scan chain.

⊞ C Cross-Cutting Problems - *(1208)*

- Weaknesses in this category can arise in multiple areas of hardware design or can apply to a wide cross-section of components.

⊞ C Physical Access Issues and Concerns *(1388)*

- Weaknesses related to physical tamper and environmental issues

Cycuity

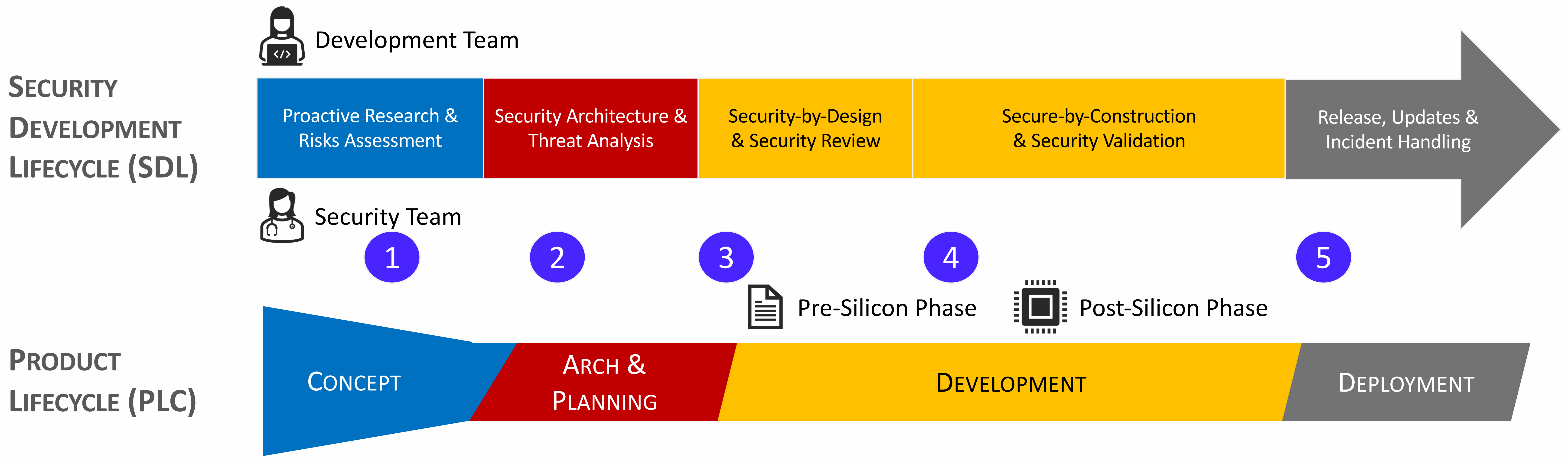
Why use CWE as a Security Metric?

- **Technology products are exposed to a rising number of software and hardware vulnerabilities over their lifetimes**
  - Over 160K vulnerabilities between 1999 to 2019
  - Over half were reported in the last 5 years
- **Exposure increases for technologies with longer product lives**
- **Majority of these vulnerabilities have been reported before**
  - Repeated mistakes suggest effective assurance practices are yet to be widely adopted
  - First step: Gain a deep understanding of common vulnerability patterns
- **CWE provides a proactive framework to assess how secure a design is**

Using CWE to Drive a Security Development Lifecycle

1. **Research** common weaknesses impacting related products to strengthen product threat models
2. **Identify** effective mitigation options per threat
3. **Educate** to avoid common design & implementation pitfalls
4. **Evaluate** relevant weakness categories to enumerate security properties & effective detection techniques
5. **Triage** and classify escapes to facilitate PSIRT handling & training, process improvements

Organizations can use CWE can be used effectively across each of these 5 stages



Development Team
Security Development Lifecycle (SDL)
Proactive Research & Risks Assessment
Security Architecture & Threat Analysis
Security-by-Design & Security Review
Secure-by-Construction & Security Validation
Release, Updates & Incident Handling
Security Team
1
2
3
4
5
Pre-Silicon Phase
Post-Silicon Phase
Product Lifecycle (PLC)
Concept
Arch & Planning
Development
Deployment

Cycuity