SECURE-IC
THE SECURITY SCIENCE COMPANY

AUTOMATING CYBER-PHYSICAL SECURITY VERIFICATION IN THE SOC DESIGN FLOW

Valentin Peltier, Product Marketing Manager

July 2025

VF2025
VERIFICATION FUTURES

AGENDA

1. INTRODUCTION
2. ABOUT PHYSICAL SECURITY
3. VERIFICATION TOOLS & AUTOMATION
4. CONCLUSION
5. SECURE-IC'S SOLUTIONS

1. INTRODUCTION

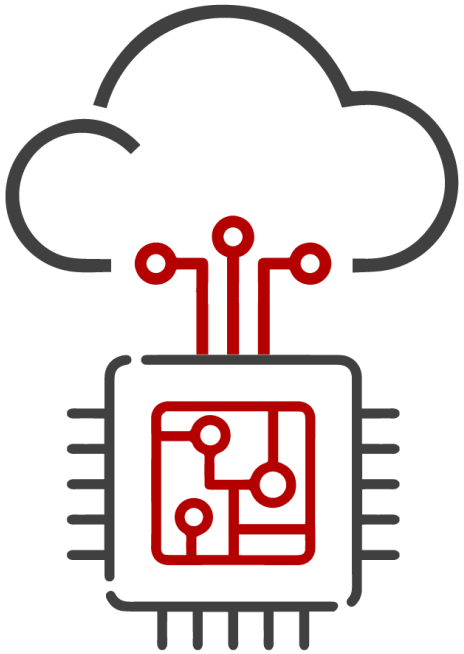
A GLOBAL LEADER IN EMBEDDED SECURITY

IoT devices being interconnected, **each and every object could be a threat for the whole network.**

Therefore, the **security of the objects or the devices with their lifecycle management is key**, and so is their data. To ensure the integrity of this data, the whole system must be secured and managed. **Trusted devices enable trusted data**.

Secure-IC partners with its clients to provide them with the best end-to-end cybersecurity solutions for embedded systems and connected objects, **from Chip to Cloud**

ONE DAY, SECURITY WILL BE WORTH MORE THAN THE DEVICES



2. ABOUT PHYSICAL SECURITY

CRYPOGRAPHY IS ROBUST, BUT…

SECURITY THREATS

Cryptography is robust but **physical attack** are here…

- **Side-Channel Analysis** (SCA)
- **Fault Injection Analysis** (FIA)
- Cache-Timing Analysis
- Reverse Engineering
- Hardware Trojan
- …

A BIT OF HISTORY

Side-Channel Analysis

- Introduced in the 1990s by Paul Kocher*, an American cryptographer
- Recover secret keys by **measuring the execution time**
- Physical attacks was born!

**Paul Kocher, 1996, Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems*

Fault Injection Analysis

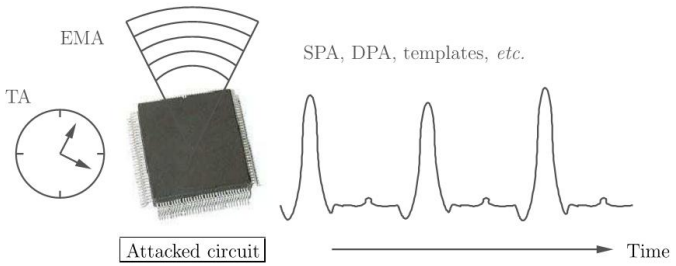
- Initially used in hardware testing & reliability engineering, observing how system respond → safety-oriented
- Evolution into a **security-focused** methodology in the 1990s**

**
- *R. Anderson & M. Kuhn, 1996, Tamper Resistance - a Cautionary Note*
- *D. Boneh, R. DeMillo, & R. Lipton, 1997, On the importance of checking cryptographic protocols for faults*
- *E. Biham & A. Shamir, 1997, Differential fault analysis of secret key cryptosystems*

PRINCIPLE

Side-Channel Analysis

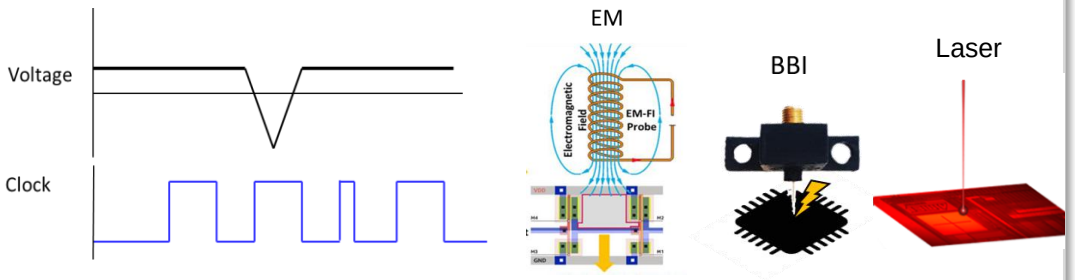
- **Measure execution activity** of sensitive data into embedded system



- All these characteristics become a potential leakage

Fault Injection Analysis

- **Disturb** embedded system **during execution**

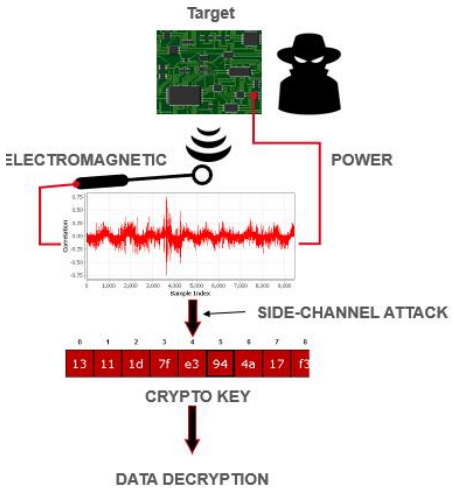


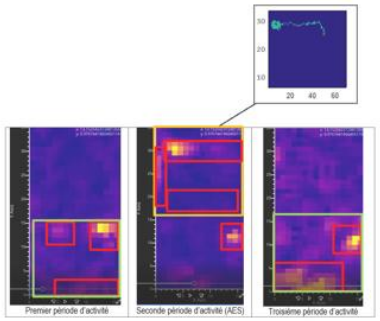
- Injecting faults at different level:
  - Physical (memory, transistors, logic gates)
  - Micro-Code (skip/jump instruction, read/write)
  - Code (IO/crypto routines, loop statement)

EXPLOITATION

Side-Channel Analysis

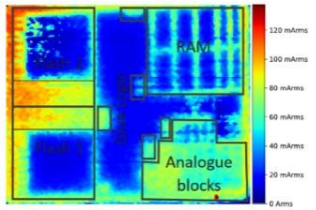
- Secret Key Recovery
- Reverse Engineering
- …





Fault Injection Analysis

- Denial of Service
- Privilege Escalation
- Secret Recovery
- Reverse Engineering
- …



COMMON WEAKNESS ENUMERATION

(2025 update in progress)

- MIHW 2021: the first list **focused on hardware security**
- Created by the Hardware CWE SIG (experts from industry, academia, and government)
- **CWE-1300 Improper Protection of Physical Side Channels**
- **CWE-1247 Improper Protection Against Voltage and Clock Glitches**
- **CWE-1332 Improper Handling of Faults that Lead to Instruction Skips**

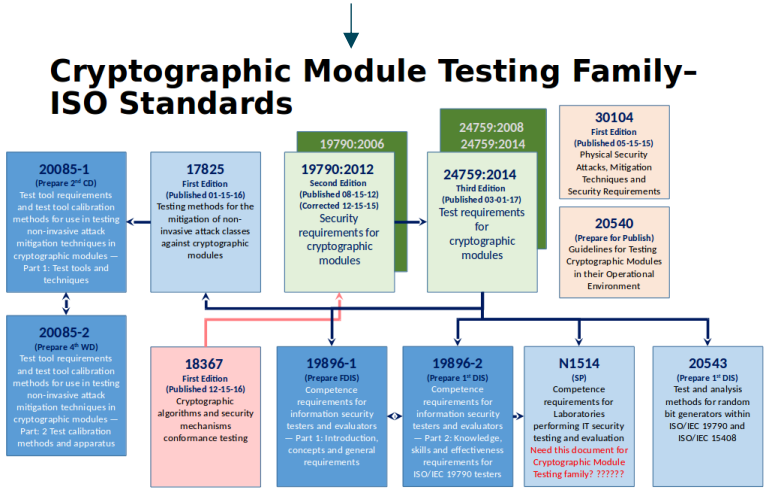
- And there are many others…
  - CWE-1319**:** Improper Protection against Electromagnetic Fault Injection
  - CWE-1384: Improper Handling of Physical or Environmental Conditions
  - …

STANDARD CERTIFICATIONS

Common Criteria & Vulnerability Assessment

Range of values* *final attack potential = identification + exploitation	TOE resistant to attackers with attack potential of:	Analyses to be carried out in Secure-IC tools (Catalyzr/Virtualyzr/Analyzr)	Corresponding AVA_VAN	Corresponding EAL
0-15	No rating	No analysis required	AVA VAN.1: vulnerability survey	EAL 1
16-20	Basic	SPA (Single trace analysis)	AVA_VAN.2: vulnerability analysis	EAL2/3
21-24	Enhanced-Basic	DPA ([1st-order] Differential analysis)	AVA VAN.3: focused vulnerability analysis	EAL 4
25-30	Moderate	High-order analysis	AVA_VAN.4: methodical vulnerability	EAL 5 (or4+) analysis
31 and above	High	All abovementioned analysis, plus: collision-analysis, stochastic analysis, mutual information analysis, template attack, machine-learning attacks, deep-learning attack	AVA_VAN.5: advanced methodical vulnerability analysis	EAL 6/7 (or 5+)

FIPS 140-3 is transitioning to ISO/IEC 19790



ISO 17825

Table 1 — Associations between non-invasive attack methods and security functions covered by this International Standard

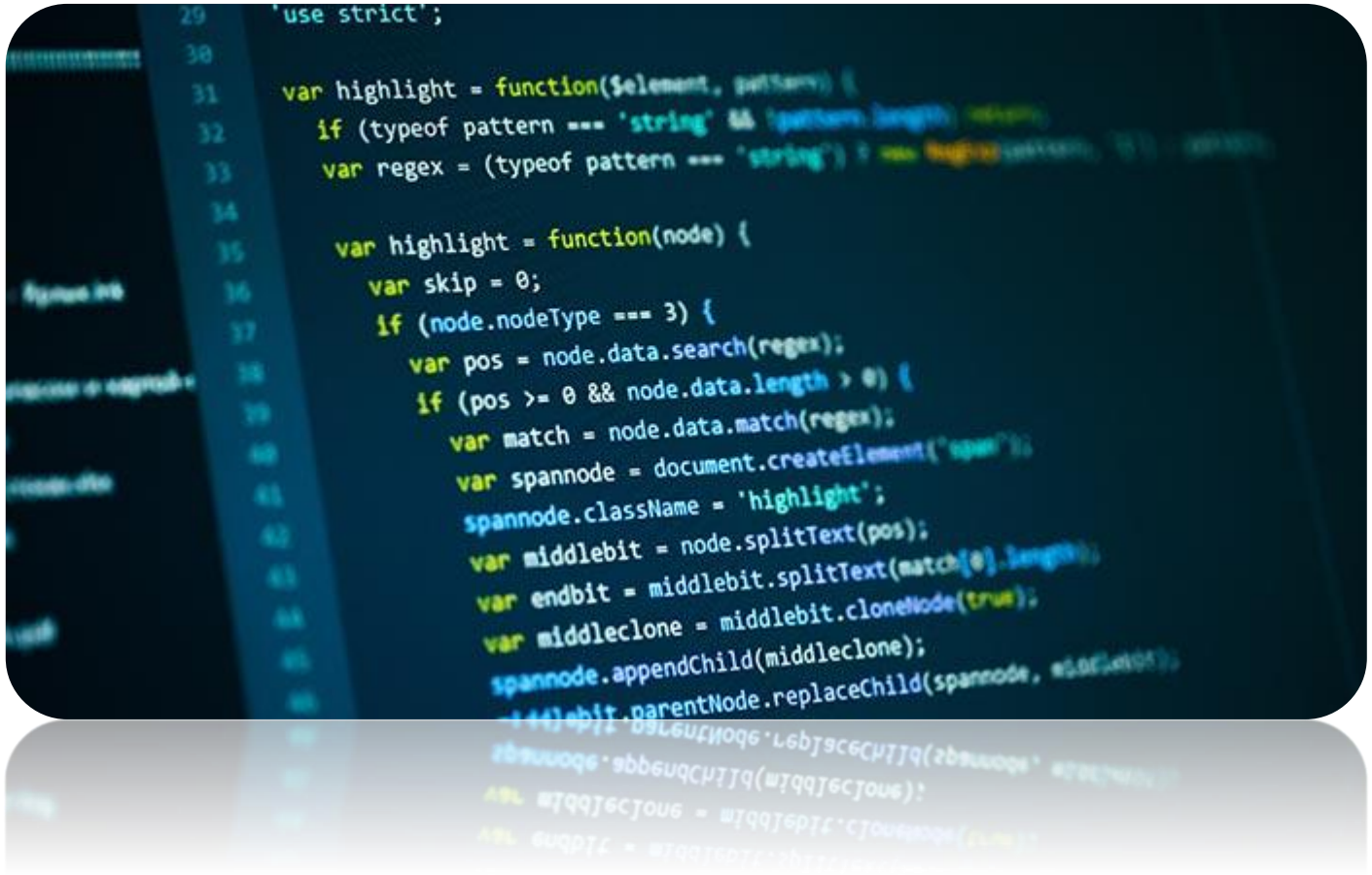
Security functions		Non-invasive attack methods		
		SPA/SEMA	DPA/DEMA	TA
Symmetric-Key	AES	A	A	A
	Triple-DES	A	A	A
	Stream Ciphers	A	A	A
Asymmetric-Key	Plain RSA (Key wrapping)	A	A	A
	RSA PKCS#1 v1.5	A	A	A
	RSA PKCS#1 v2.1	NA	NA	A
	DSA	A	A	A
	ECDSA	A	A	A
Hashing mechanisms	SHA	A	NA	NA
RNG and RBG	Deterministic	A	NA	NA
	Non-deterministic	A	NA	NA

SECURE-IC
THE SECURITY SCIENCE COMPANY

3. VERIFICATION TOOLS & AUTOMATION

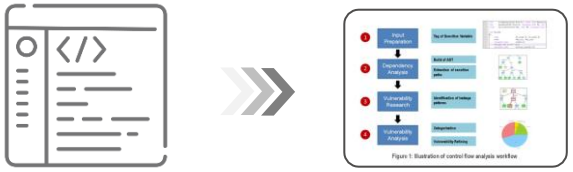
ALONG DESIGN FLOW & V-CYCLE

SOFTWARE LAYER VERIFICATION

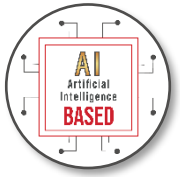


PRE-SILICON VERIFICATION (SOFTWARE)

SCA at Static Level



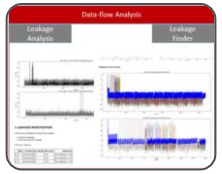
- **Code source (C/C++) verification**



- Timing vulnerability detection & warning
  (conditional branch, array indexation, …)

$$\begin{array}{l}\textbf{for } i = k-1 \textbf{ to } 0 \textbf{ do} \\ \quad a \leftarrow a^2 \mod n \\ \quad \textbf{if } d_i = 1 \textbf{ then} \\ \quad\quad a \leftarrow a \times m \mod n \\ \textbf{return } a\end{array}$$

SCA at Dynamic Level



- **Binary execution monitoring (emulation)**

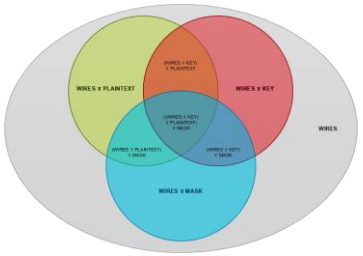
- Register dumping to rebuild activity
  (EM/Power traces)
- Cryptanalysis methodologies
  (**NICV**, T-TEST, **TVLA**, **CPA**, DPA, SPA…)

- **Leakage location on design code**

File	Function	Line	Register	Sample
aes-challenge-c.c	ark_sb_rnd	95	eflags	207
aes-challenge-c.c	ark_sb_rnd	95	eflags	211
aes-challenge-c.c	ark_sb_rnd	95	eflags	274

PRE-SILICON VERIFICATION (SOFTWARE)

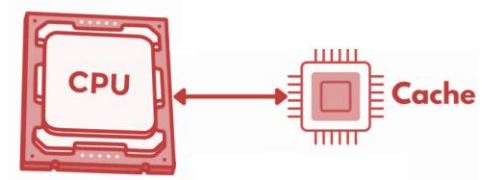
Masking Scheme Checker



- **Verify the masking protection coverage**

- Wires & mask dependencies
- Correlation on *protected* vs. *unprotected* databases

Cache-Timing Analysis



- **Mesures memory access times (cache vs. RAM)**

- Differences leak sensitive data
- e.g. Time taken during AES encryption is key dependent

Binary Analysis

- **Firmware analysis for security checkup**

- System (filesystem, system users & utilities)
- Sensitives files (certificates, private keys)
- Security level (system calls, code practice…)

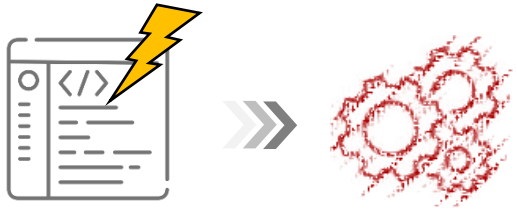
Crypto Misuse

- **Cryptographic integration checkup**

- Detect cryptography misuse in a crypto API
- Rule-based analysis engine
- e.g. PKCS11, OpenSSL, Secure-IC iSE neo…

PRE-SILICON VERIFICATION (SOFTWARE)

FIA at Static Injection Level



- **Code source (C/C++) modification**

- Manual modification before compilation
  (code instruction, global variable)
- Output execution analysis

FIA at Dynamic Injection Level

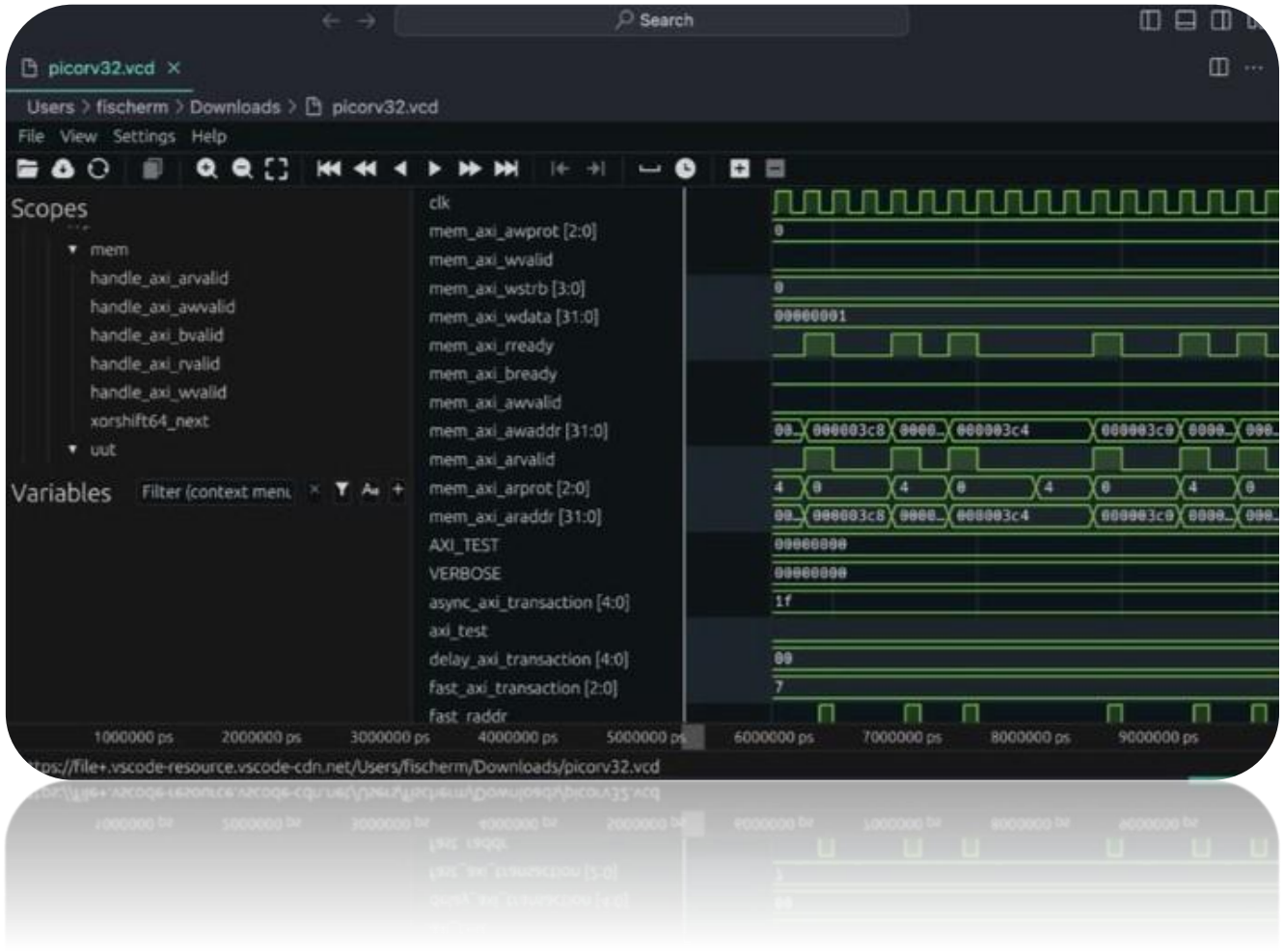
- **Binary execution disturbance (emulation)**

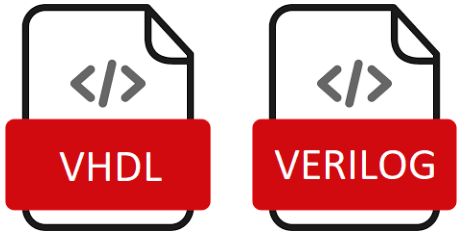
- Automated modification during emulation
  (register corruption, instruction jump/bypass)
- Output execution analysis

- **Leakage location on design code**

File	Function	Line	Register	Sample
aes-challenge-c.c	ark_sb_rnd	95	eflags	207
aes-challenge-c.c	ark_sb_rnd	95	eflags	211
aes-challenge-c.c	ark_sb_rnd	95	eflags	274

HARDWARE LAYER VERIFICATION



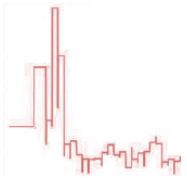


PRE-SILICON VERIFICATION (HARDWARE)

- **Hardware design simulation**
  (VHDL/Verilog) (RTL/PS/PR)
  - Value Change Dump monitoring to rebuild activity (EM/Power traces)

RUN TESTBENCH

VCD



TRACES (**NO NOISE**)

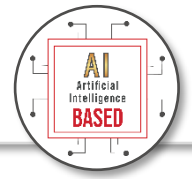
ANALYSIS

SCA Leakage Detection

- **Critical Security Parameter monitoring**
- Vulnerability detection & warning
  (NICV, **T-TEST**)

SCA Leakage Exploitation

- **Secret key recovery analysis**
- Cryptanalysis methodologies
  (**TVLA**, **CPA**, DPA, SPA…)



- **Leakage location on design code**
  High Level (modules)
  Low Level (signals & registers)
- **Countermeasure checker** (masking)

PRE-SILICON VERIFICATION (HARDWARE)

- **Hardware design simulation**
  (VHDL/Verilog) (RTL/PS/PR)

FIA Clock Glitch

- **Glitch fault model applied to the clock**

➢ CLK signal modification

FIA ElectroMagnetic

- **EM fault models applied to any register/signal**

➢ Bit freeze, Stuck at 1 (or 0)

FIA Laser

- **Laser fault models applied to any register/signal**

➢ Bit flip, Bit set, Bit reset

➢ **Cryptanalysis: secret key recovery**
(Differential Fault Analysis)

AUTOMATED PROCESS

Vulnerability Assessment

PHYSICAL SECURITY

CONFIG

NODES

SCA FIA

STIMULUS

HDL

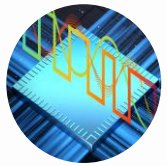
VHDL VERILOG

DESIGN
RTL / PS / PR

SoC

RUN TESTBENCH

EDA-SIMULATOR



EDA-EMULATOR

VCD

COMPUTING

DATA EXTRACTION

TRACES

ANALYSIS

AVA_VAN (1~5)

GO / NO GO

Cryptanalysis
- Leakage Detection (NICV/T-test)
- Leakage Exploitation (Secret Key Recovery)

Source code mapping
- High Level (sub-modules)
- Low Level (signals and registers)

Countermeasure checker (masking)

4. CONCLUSION

TOWARDS A 360º SECURITY VERIFICATION STRATEGY

- **Security must be verified—not just designed**
  - Side-Channel and Fault Injection analysis are critical threats requiring proactive mitigation from RTL to silicon.
- **Automation enables true lifecycle coverage**
  - Embedding physical analysis (SCA, FIA) into standard verification flows ensures early detection and validation throughout the SoC development process.
- **Accelerate compliance and reduce risk**
  - Supports faster certification (CC, ISO, CAVP, FIPS 140-3) and reduces manual effort for design teams up to security sign-off.

5. SECURE-IC'S SOLUTIONS

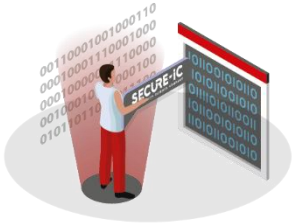
3 TOOLS FOR SECURITY VERIFICATION ALONG DEVICE LIFEFCYCLE

DESIGN LIFE CYCLE

- **DEVICE** LAYER
- **SOFTWARE** LAYER
- **HARDWARE** LAYER

ANALYZR™

CATALYZR™



VIRTUALYZR™

- **REAL DEVICE** EVALUATION
- **SOFTWARE** VERIFICATION
- **PRE-SILICON** VERIFICATION

SECURITY LIFE CYCLE

DESIGN FOR SECURITY SOLUTIONS

	FORMAL VERIFICATION		SIMULATION	EMULATION
DFS in depth				
Functional Analysis	Jasper™ C/C++ Apps (C/C++)	Jasper™ RTL Apps (RTL)	Xcelium™ Apps (RTL/PS/PR)	Palladium™ Apps (RTL/PS/PR)
Security Analysis Data Flow based		SPV App (RTL)		
Physical Security Analysis Data Manipulation based (Common Criteria, FIPS, ISO/IEC)			Secure-IC Virtualyzr™ (RTL/PS/PR)	Secure-IC Virtualyzr™ (RTL/PS/PR)
				Secure-IC Catalyzr™ EMU (C/C++)

CATALYZR™
Software design layer
C/C++

VIRTUALYZR™
Hardware design layer
RTL/PS/PR

SECURE-IC
THE SECURITY SCIENCE COMPANY

THANK YOU FOR YOUR ATTENTION

CONTACTS

EMEA	sales-EMEA@secure-IC.com
APAC	sales-APAC@secure-IC.com
CHINA	sales-CHINA@secure-IC.com
JAPAN	sales-JAPAN@secure-IC.com
TAIWAN	sales-TAIWAN@secure-IC.com
AMERICAS	sales-US@secure-IC.com

FOLLOW US ON SOCIAL MEDIA