

# Happy Birthday Sancus! – Lessons from 10 Years of Maintaining a Trusted Computing Research Prototype

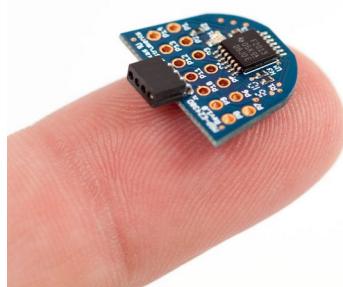
Jo Van Bulck, Frank Piessens

March 24, 2023, DRADS

# What is Sancus?

*A crash course introduction*

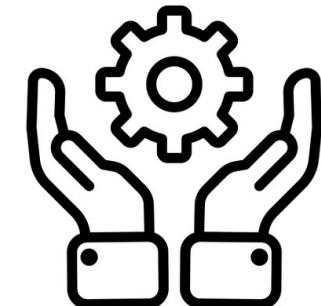
# Sancus: Lightweight trusted computing for the IoT



+



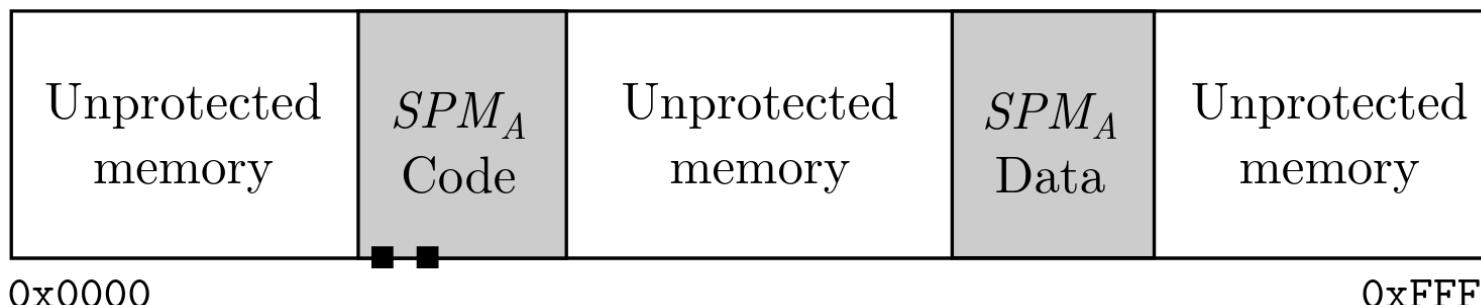
+

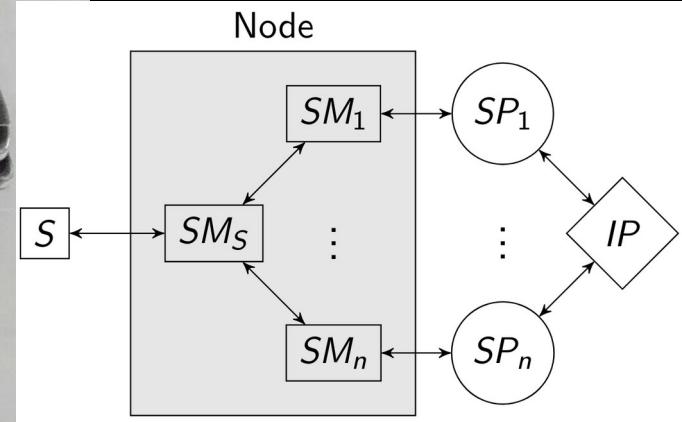
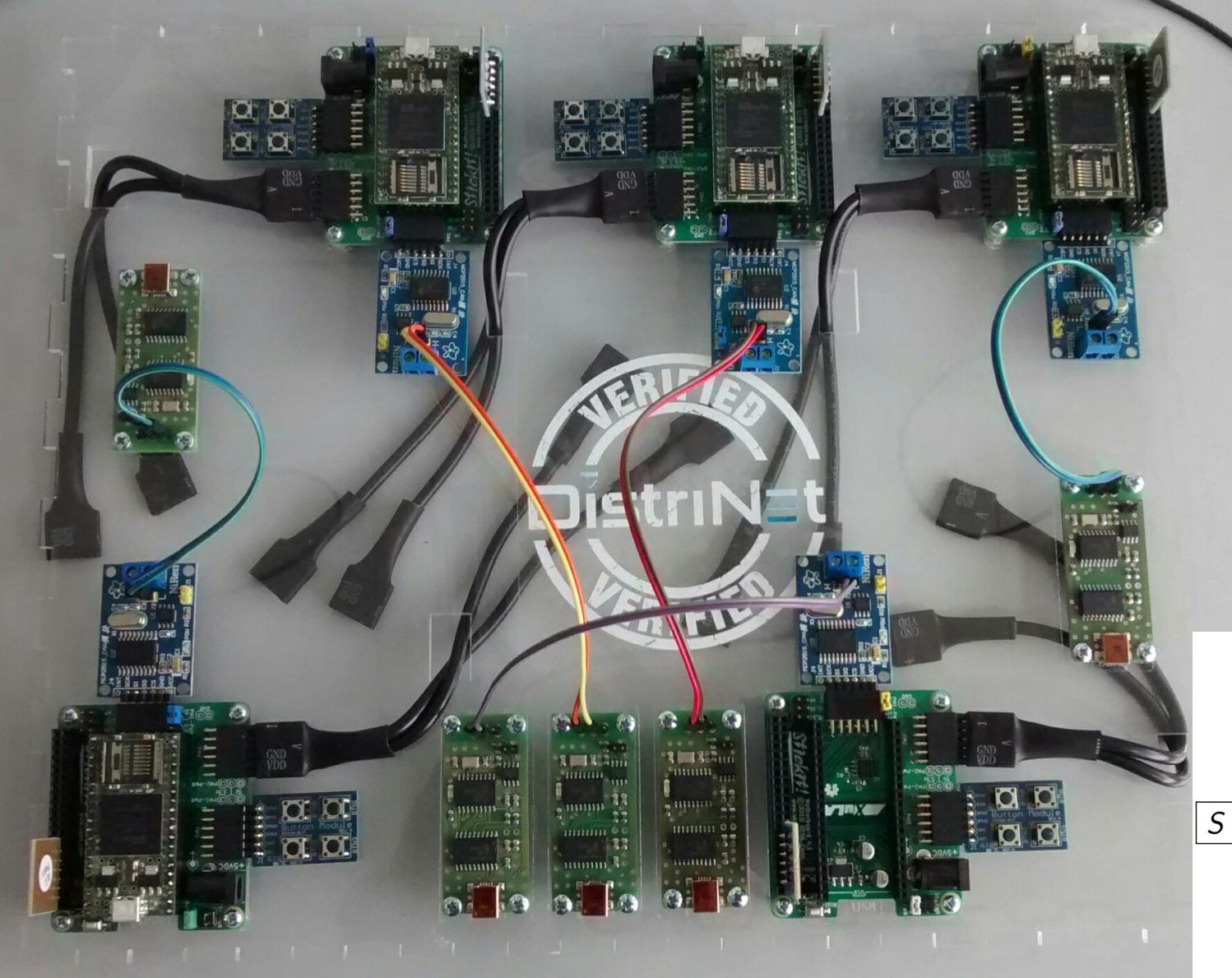


**OpenMSP430 CPU extensions  
for isolation + attestation**

**LLVM compiler pass**

**Support software  
“operating system”**



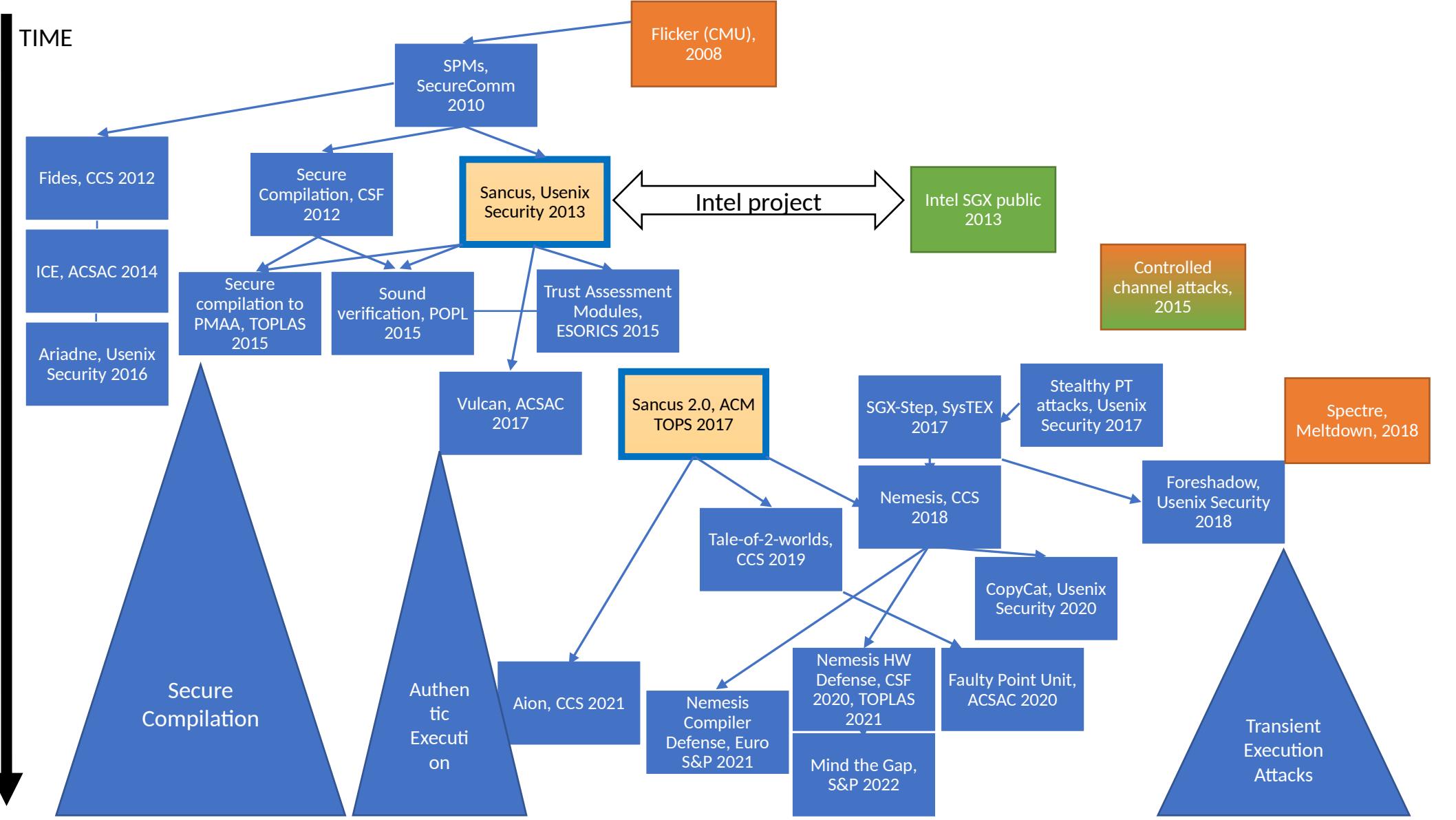


# Research landscape

*A retrospective of the bigger picture*

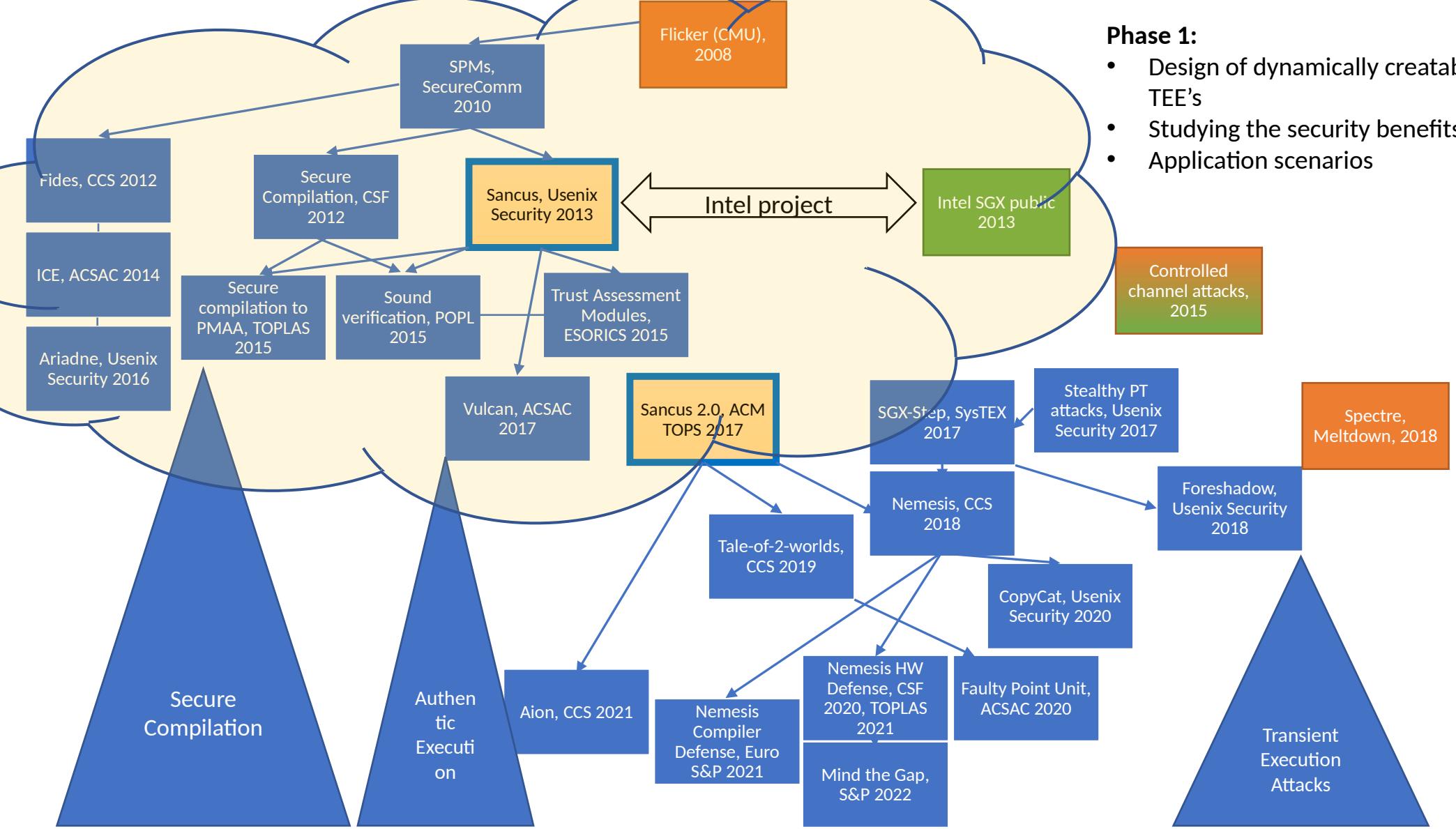
# Historical context

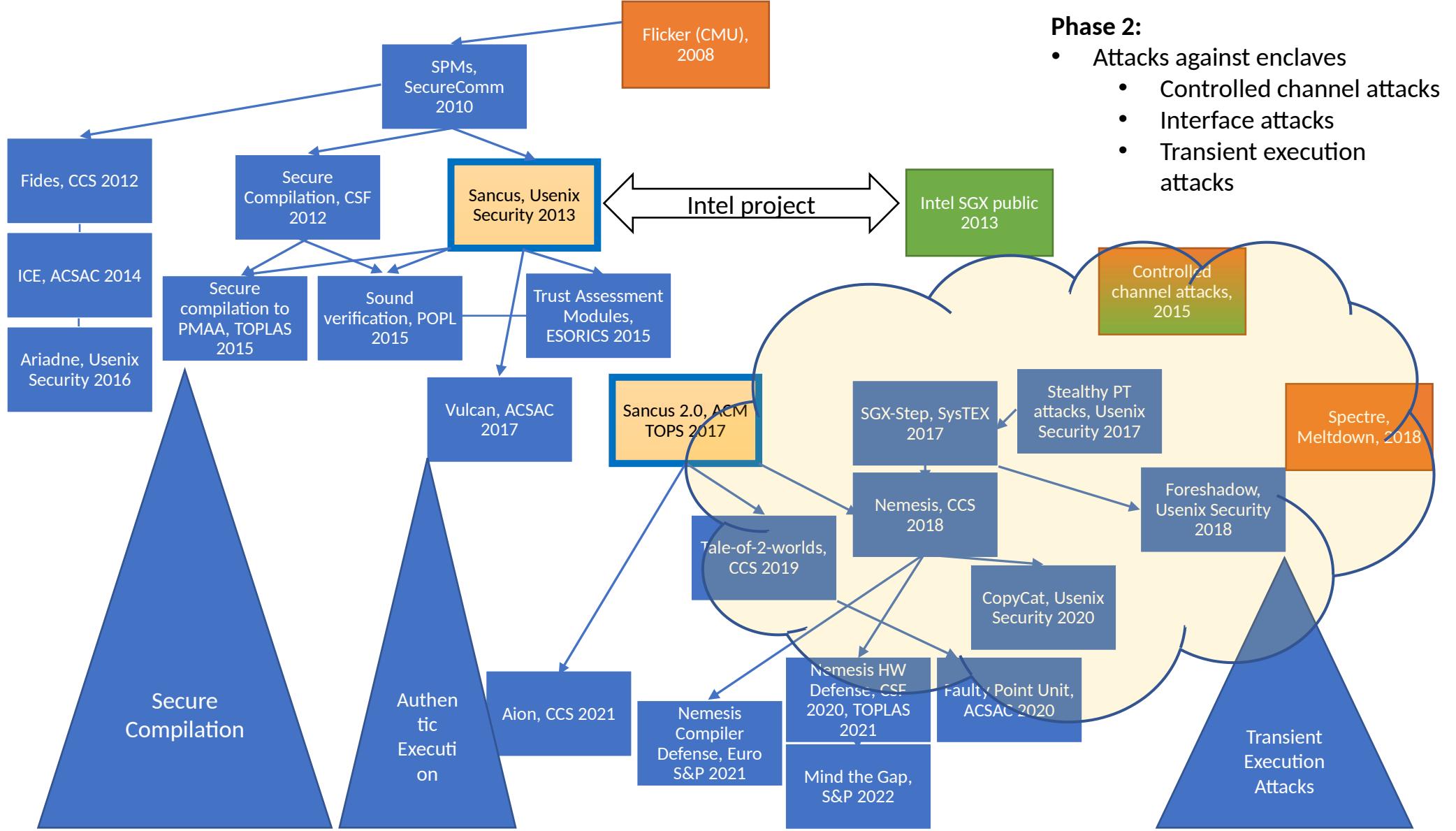
- Objective of this part: some historical context, and some lessons learned from the supervisor perspective
- No technical details
- Disclaimer:
  - This is an account of history as I remember it, no correctness guarantees ^^
  - It only covers efforts that I was directly involved in, and misses other interesting Sancus-related work, e.g.:
    - From COSIC, work like Soteria
    - From the DistriNet NES task force, work like the Security MicroVisor and follow-up work
    - ...



## Phase 1:

- Design of dynamically creatable TEE's
- Studying the security benefits
- Application scenarios

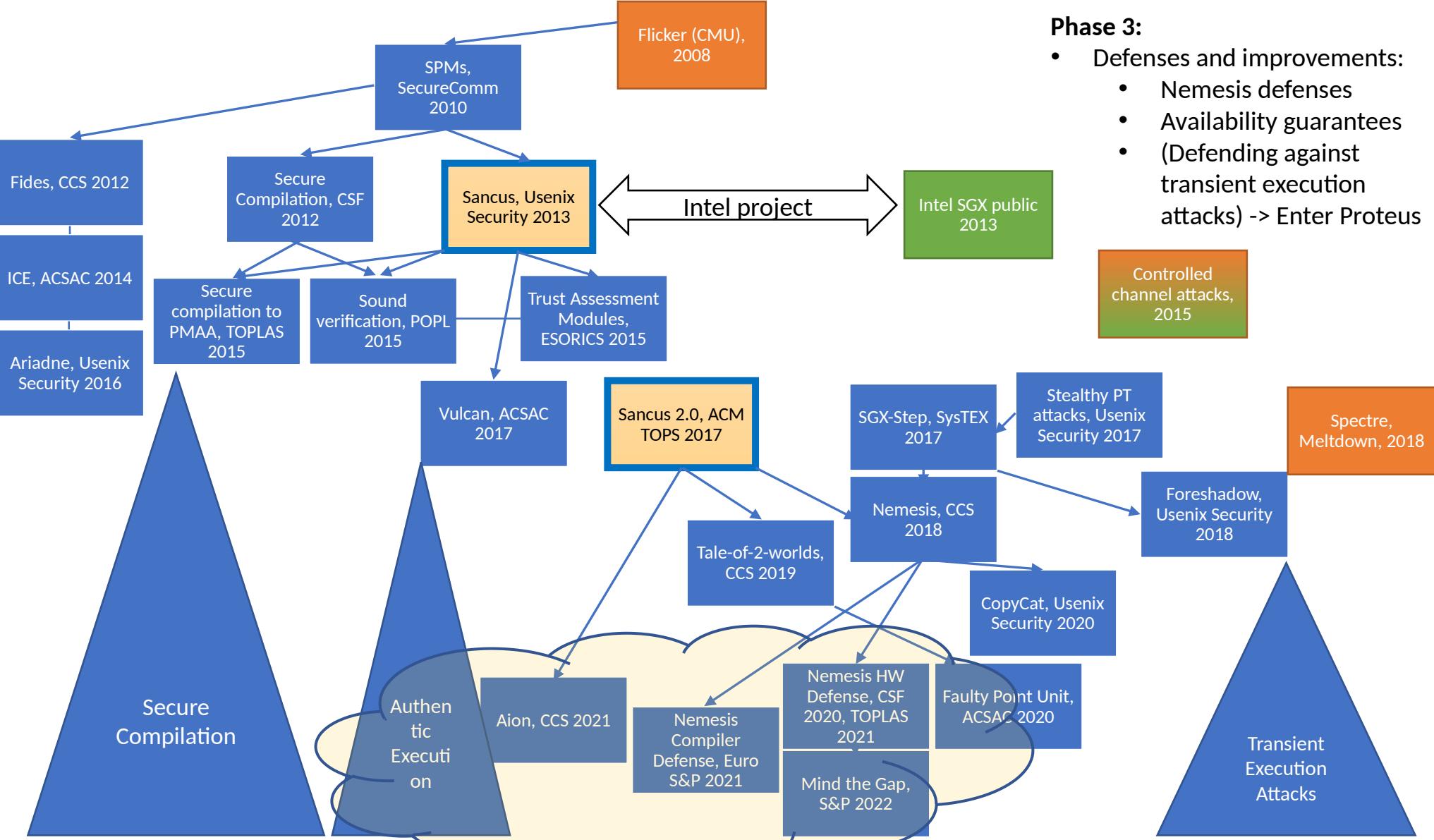




- Phase 3:**
- Defenses and improvements:
    - Nemesis defenses
    - Availability guarantees
    - (Defending against transient execution attacks) -> Enter Proteus

Controlled channel attacks, 2015

Spectre, Meltdown, 2018



## Phase 4: The Immediate Future

- Proteus and RISC-V
- Defenses

channel attacks,  
2015

Spectre,  
Meltdown, 2018

Fides, CCS 2012

Secure  
Compilation, CSF  
2012

Sancus, Usenix  
Security 2013

Intel project

Intel SGX public  
2013

ICE, ACSAC 2014

Secure  
compilation to  
PMAA, TOPLAS  
2015

Sound  
verification, POPL  
2015

Trust Assessment  
Modules, ESORICS 2015

Ariadne, Usenix  
Security 2016

Vulcan, ACSAC  
2017

Sancus 2.0, ACM  
TOPS 2017

SGX-Step, SysTEX  
2017

Stealthy PT  
attacks, Usenix  
Security 2017

Foreshadow,  
Usenix Security  
2018

Secure  
Compilation

Authen-  
tic  
Execu-  
tion

Transient  
Execution  
Attacks

Tale-of-2-worlds,  
CCS 2019

Nemesis, CCS  
2018

CopyCat, Usenix  
Security 2020

Nemesis HW  
Defense, CSF  
2020, TOPLAS  
2021

Faulty Point Unit,  
ACSAC 2020

Mind the Gap,  
S&P 2022

Proteus

Aion, CCS 2021

Nemesis  
Compiler  
Defense, Euro  
S&P 2021

uArch Profiling,  
Euro S&P 2023

CheriTrEE

AMi, S&P 2024

Prospect, Usenix  
Security 2023

# Some lessons learned

- Key PhD theses have been the backbone of this research line:
  - **Raoul Strackx**, Security Primitives for Protected-Module Architectures Based on Program-Counter-Based Memory Access Control, 2014
  - **Job Noorman**, Sancus: A Low-Cost Security Architecture for Distributed IoT Applications on a Shared Infrastructure, 2017
  - **Jo Van Bulck**, Microarchitectural Side-Channel Attacks for Privileged Software Adversaries, 2020

(Corollary: role of the supervisor is limited ^^ )
- Stable / mature / well-maintained prototypes matter, for defense and attack, e.g.:
  - Sancus
  - SGX-Step
- Interactions with the broader community have been essential:
  - Academia:
    - Worldwide: Flicker, controlled-channel attacks, transient execution attacks
    - Within DistriNet and KULeuven:
      - VeriFast and PLSIG
      - Cosinet
  - Industry: Intel SGX, RISC-V

# Elements of success?

*Do's and don't for long-lived research projects*

**IF YOU'RE OLD AND WISE TODAY**



**YOU MUST HAVE BEEN  
YOUNG AND STUPID ONCE.**



## Key #1: Gather a research team

---



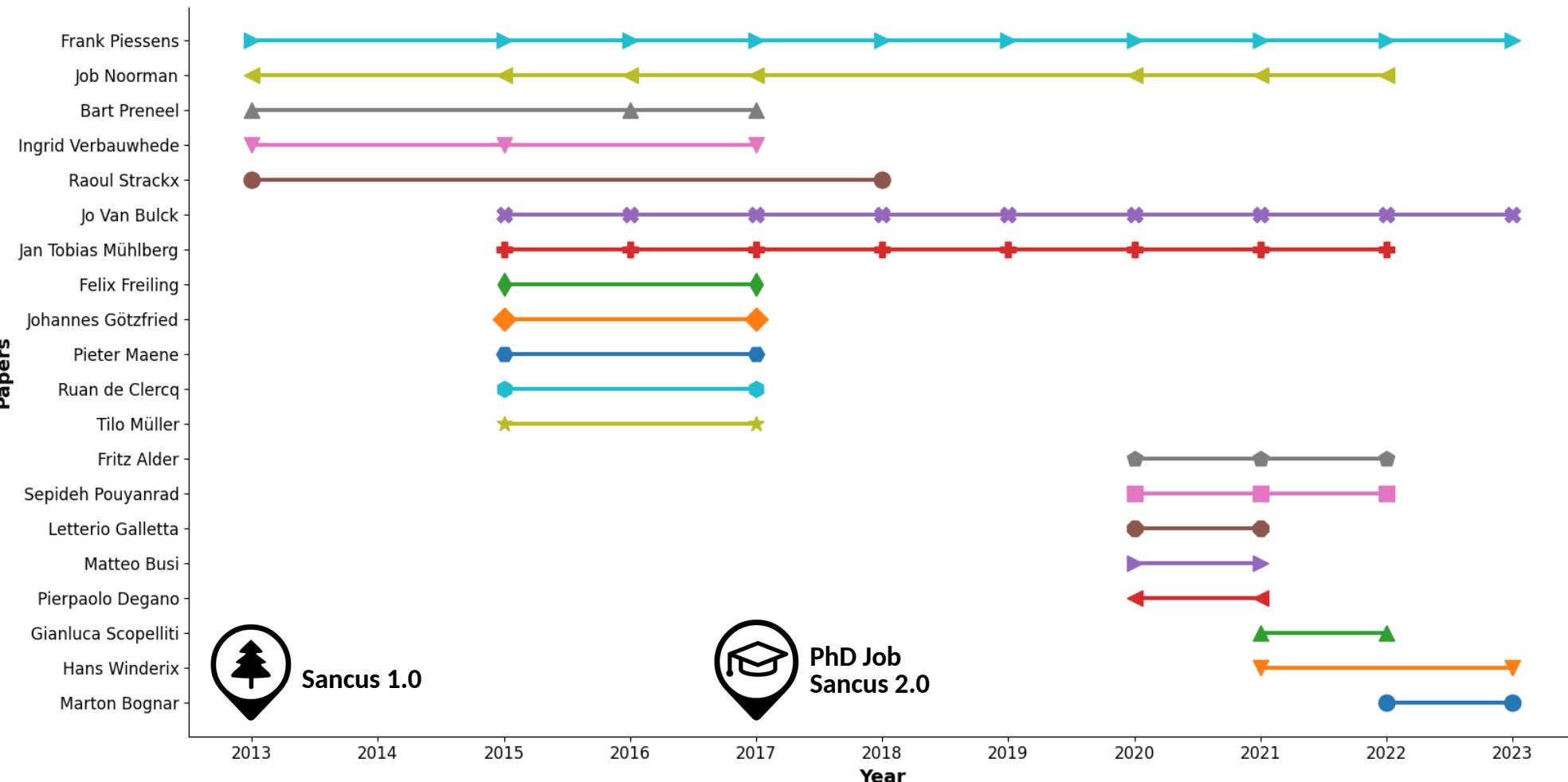
# Sancus collaboration in numbers

- **58 unique authors:** 18 DistriNet, 7 COSIC, 18 ext, 13 students
- **Inclusive:** Prof. – Postdocs – PhDs – Msc/bachelor students
- **Continuity:** 2012 – 2022 >> single PhD trajectory(!)

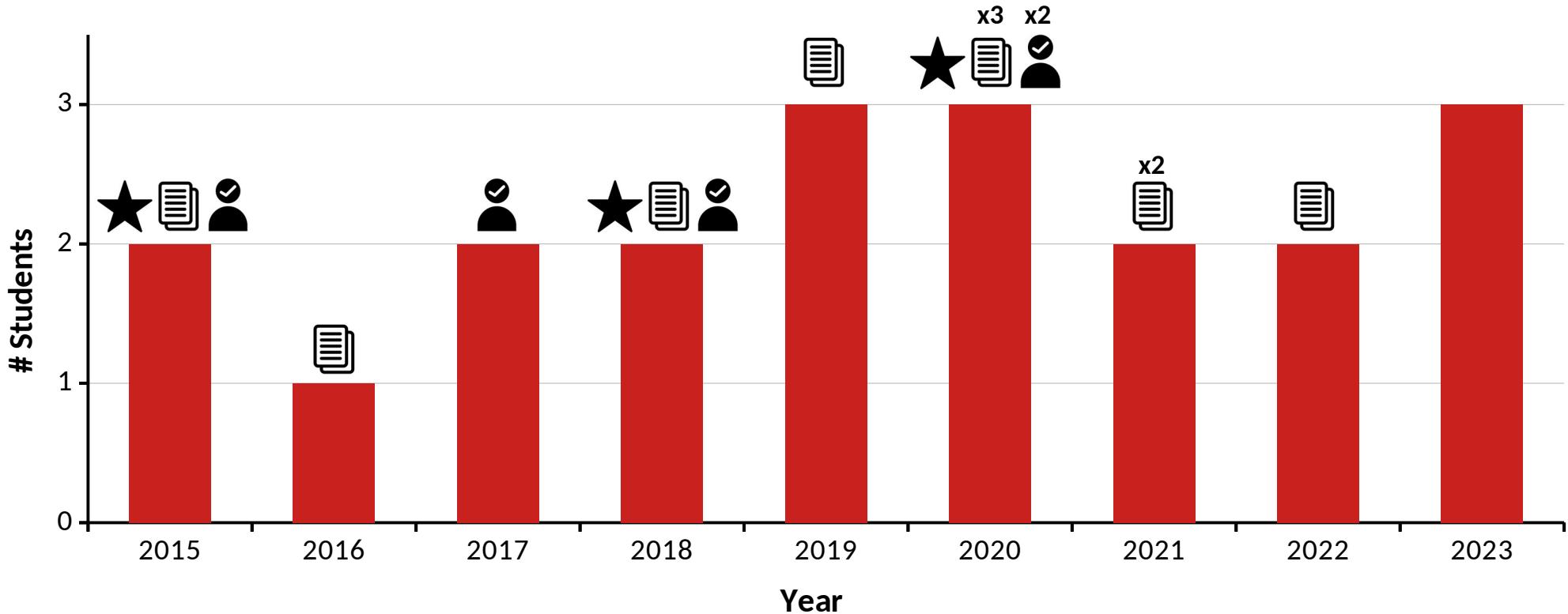


- Do combine **expertise** (hardware, software, etc.)
- Do form **sub-teams**; don't always involve everyone
- Do provide **continuity**

# Sancus team continuity: Authors >1 paper over time



# Sancus master thesis projects



- Overall: 20 students, 4 awards, 10 publications, 5 hires
- Do formulate **concrete, well-sscoped** topics; invest in **mentoring**



## Key #2: Find a relevant niche

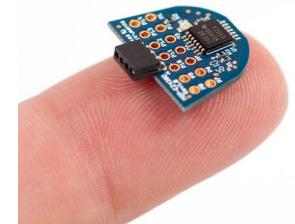
---

*“Embedded-systems security is,  
for lack of a better word, a mess.”*

– John Viega & Hugh Thompson (S&P'12)

# Sancus: Low-cost IoT enclaves with a zero-software TCB

- **Embedded:** Small **16-bit CPU** w/o existing security
- **Hardware-software co-design:** **Zero-software TCB**

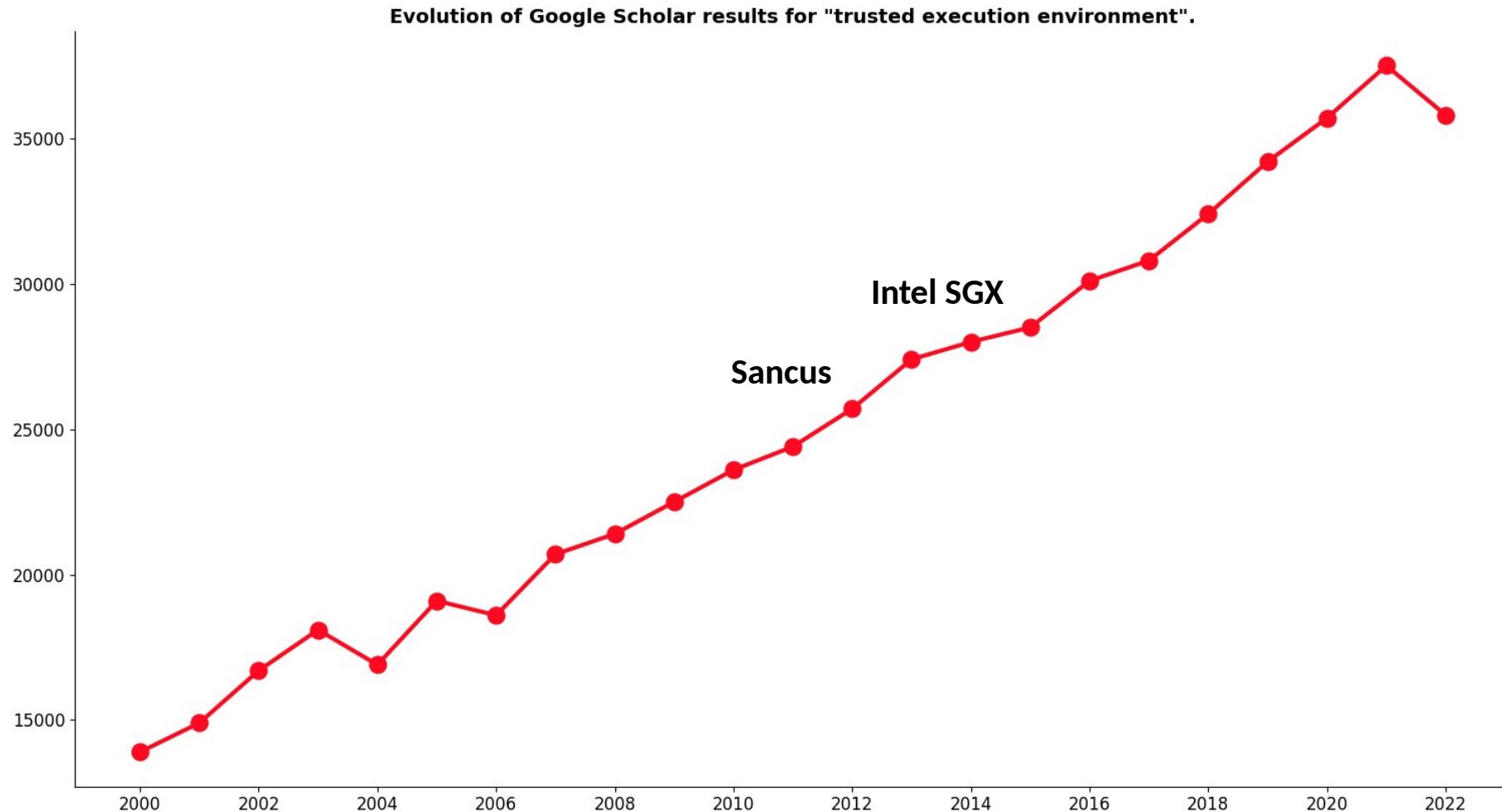


- **Full system stack:** Hardware, compiler, OS, App
- Relevant playground: ~ Real-world **Intel SGX(!)**



**Do** find a **relevant niche**, but stay connected to the **bigger picture...**

# The bigger picture: The rise of trusted execution

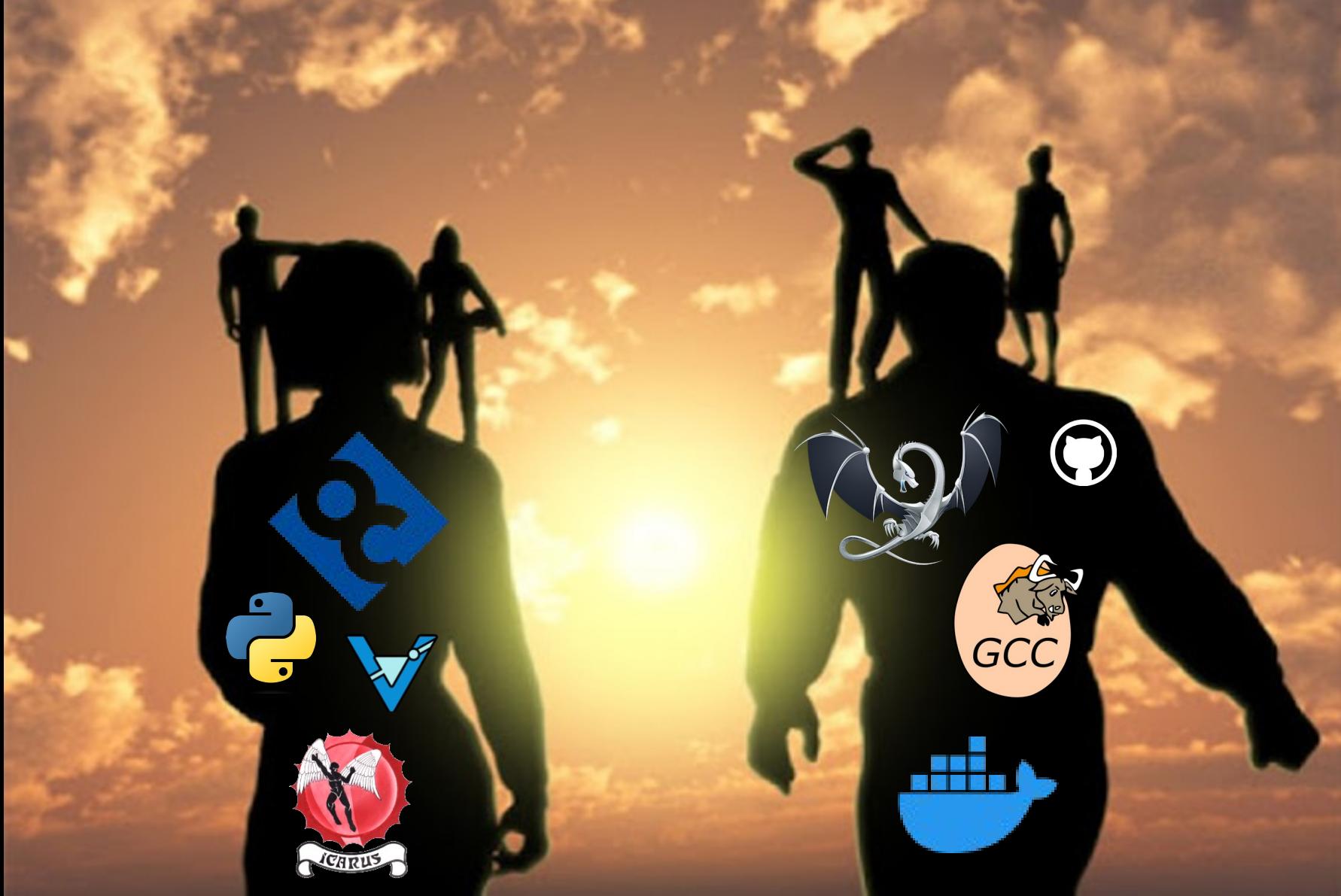




## Key #3: Open source

---





# Sancus: Open-source artifacts for reproducible science

- No commercialization/patents; FOSS licenses
- Limit **dependencies**: e.g., LLVM <> GCC
- **Upstream** eagerly: Avoid dead forks...
  - 2012-2017: Public **tarballs** + private dnetcode
  - 2017: Move to public **GitHub** organization



The screenshot shows the GitHub profile for the "Sancus" repository. It features a large icon of a Spartan helmet. The repository name "Sancus" is displayed prominently, along with a description: "A Lightweight Trusted Execution Environment for Secure IoT Devices". Below this, it shows 3 followers, the location "imec-Distrinet, KU Leuven, Belgium", and a link to the software at <https://distrinet.cs.kuleuven.be/software/>. At the bottom, there are navigation links for Overview, Repositories (16), Projects, Packages, Teams, People (8), and a Home button.

*"The most important book about technology today,  
with implications that go far beyond programming."*  
— Gary Kawasaki

Revised & Expanded

## THE CATHEDRAL & THE BAZAAR

MUSINGS ON LINUX AND OPEN SOURCE  
BY AN ACCIDENTAL REVOLUTIONARY

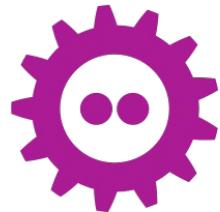
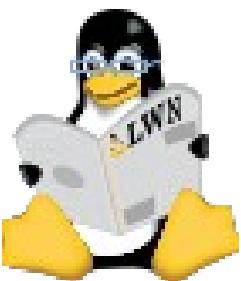


ERIC S. RAYMOND

WITH A FOREWORD BY BOB YOUNG, CHAIRMAN & CEO OF RED HAT, INC.

*“A project based on open-source building blocks  
and free-software ethos [...] should be lauded  
and considered by anyone [...]”*

– Mischa Spiegelmock, LWN.net, 2018

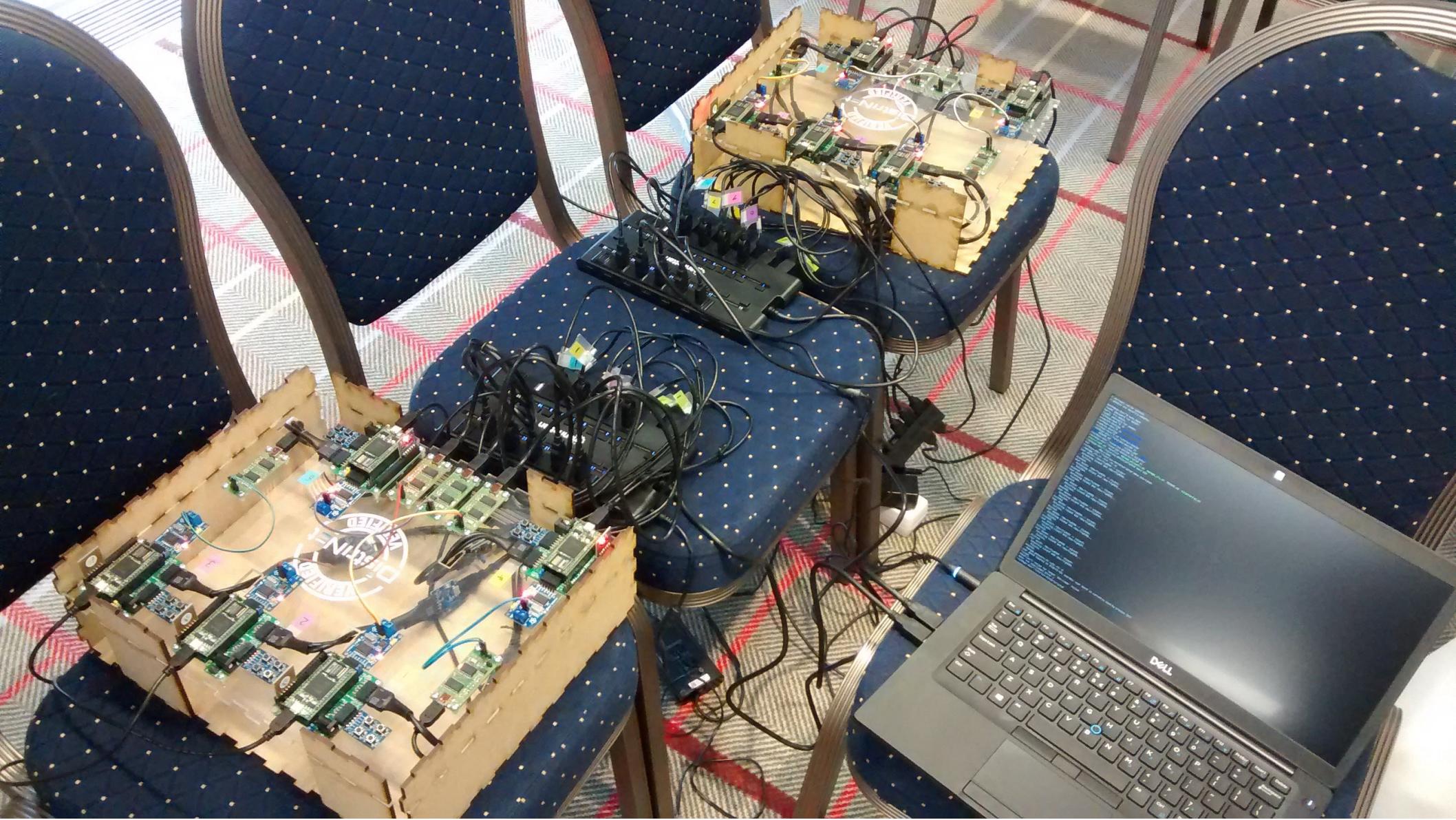




## **Key #4: Build usable systems**

---





# Building usable systems

- Large **engineering effort** ↔ minimal publication effort
- Simulators and test frameworks
- Continuous integration
- Tutorial [DSN'18] → VulCAN [ACSAC'20]



*"I'm happy to say that the evaluation worked flawlessly – great job!"*





## Key #6: Science communication

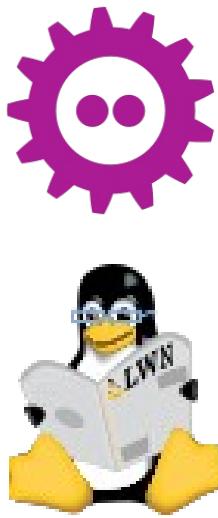
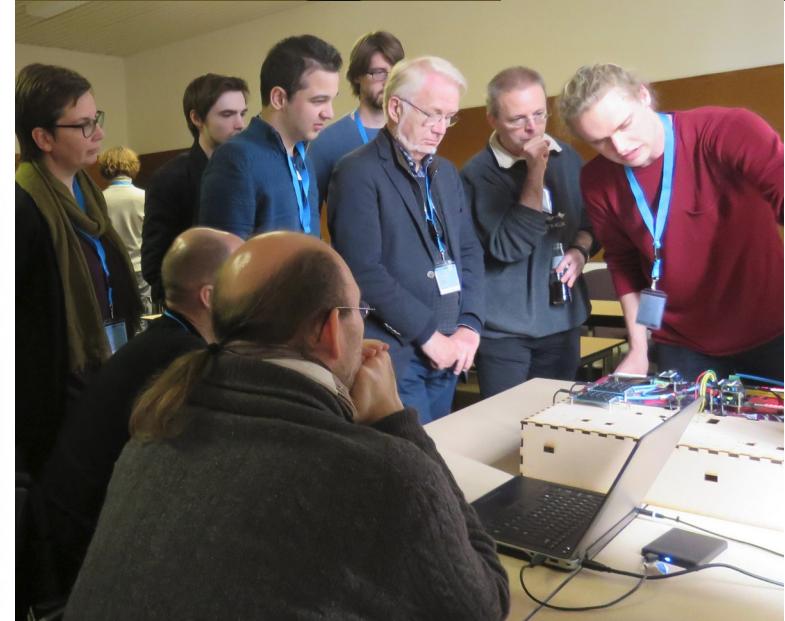
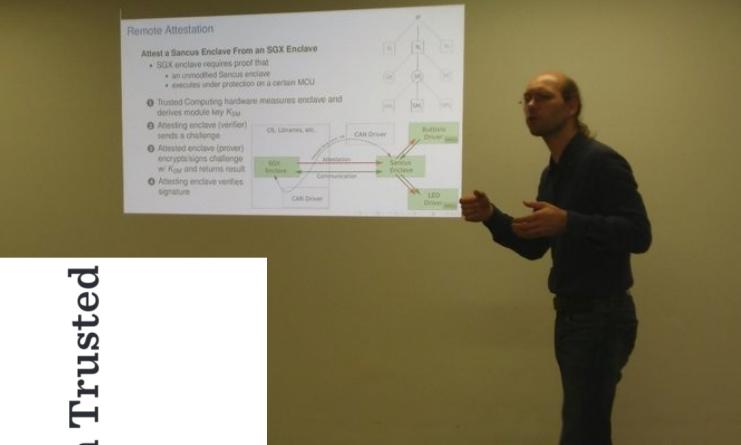
---

# AUTOMOTIVE COMPUTING



## Reflections on Post-Meltdown Trusted Computing A Case for Open Security Processors

JAN TOBIAS MÜHLBERG AND JO VAN BULCK



# Sancus: Lightweight and Open-Source Trusted Computing for the IoT

[View on GitHub](#)[Watch a demo](#)[Explore Research](#)

**“** We do have problems with security, ones that need to be dealt with, not only with changes to software toolchains but also to the underlying hardware.

—Rik Farrow *USENIX ;login:*



## SOFTWARE ISOLATION

Outside software cannot read or write a protected module's runtime state. A module can only be called through one of its designated entry points.



## SECURE COMMUNICATION

Sancus safeguards the authenticity, integrity, and freshness of all traffic between a protected module and its remote provider.



## LIGHTWEIGHT CRYPTOGRAPHY

A minimalist cryptographic hardware unit enables low-overhead symmetric key derivation, authenticated encryption, and hashing.



## SECURE I/O

Secure driver modules have exclusive ownership over memory-mapped I/O peripheral devices, and can implement software-defined access control policies.



## SOFTWARE ATTESTATION

Remote or local parties can verify at runtime that a particular software module has been isolated on a specific node without having been tampered with.



## BACKWARDS COMPATIBILITY

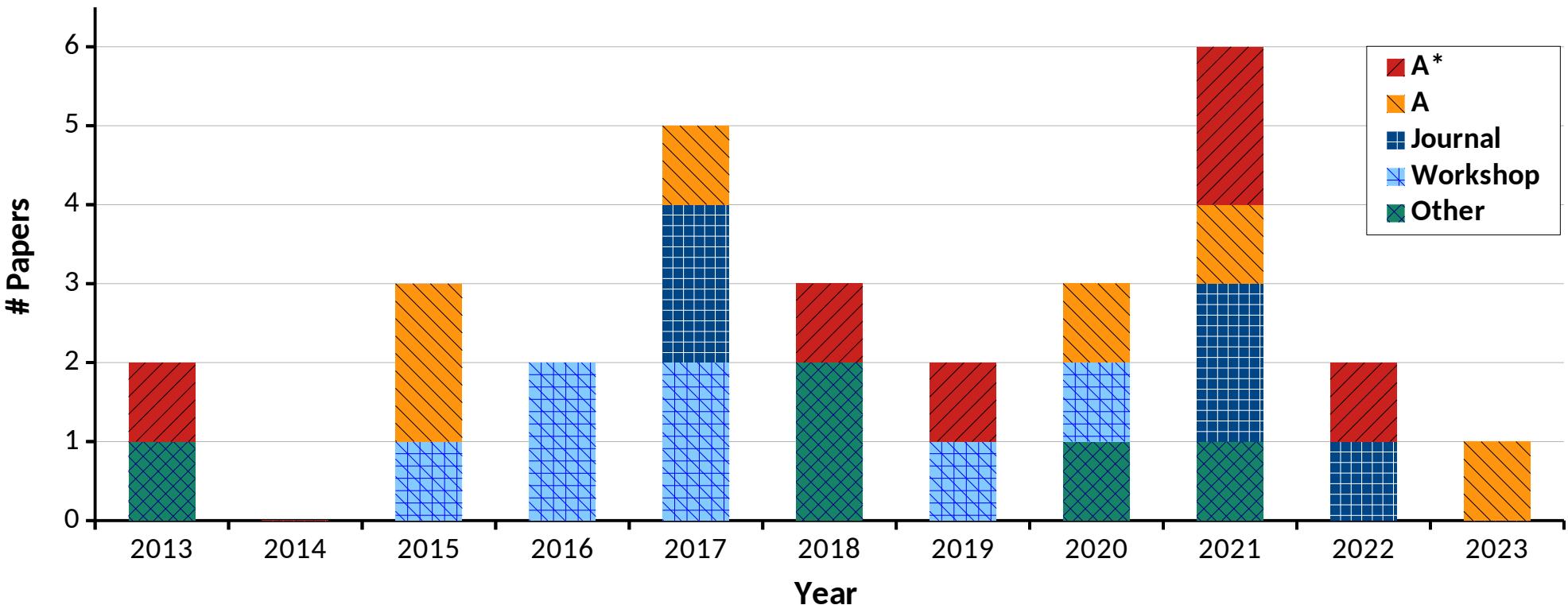
Legacy applications continue to function as expected; critical components can be migrated gradually into Sancus-protected modules.



## Key #7: When to publish

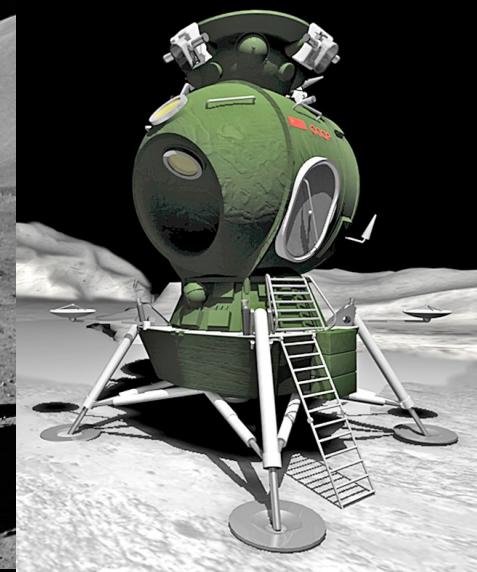
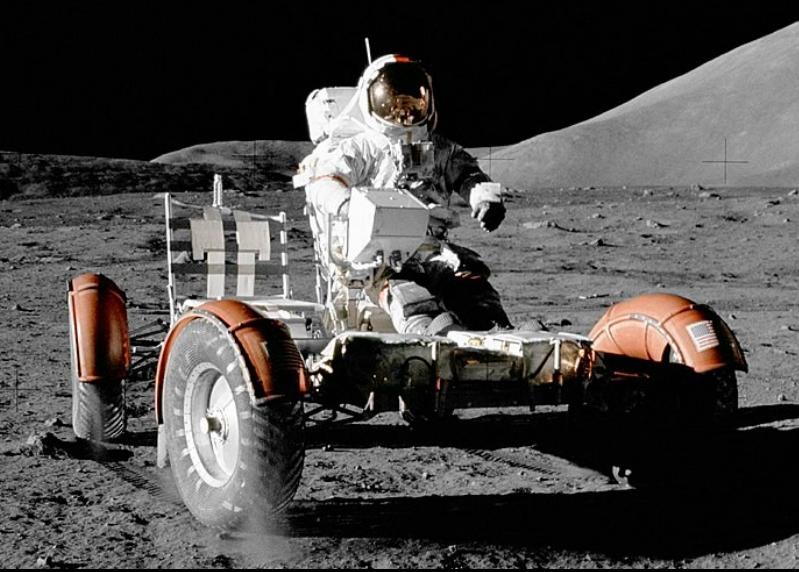
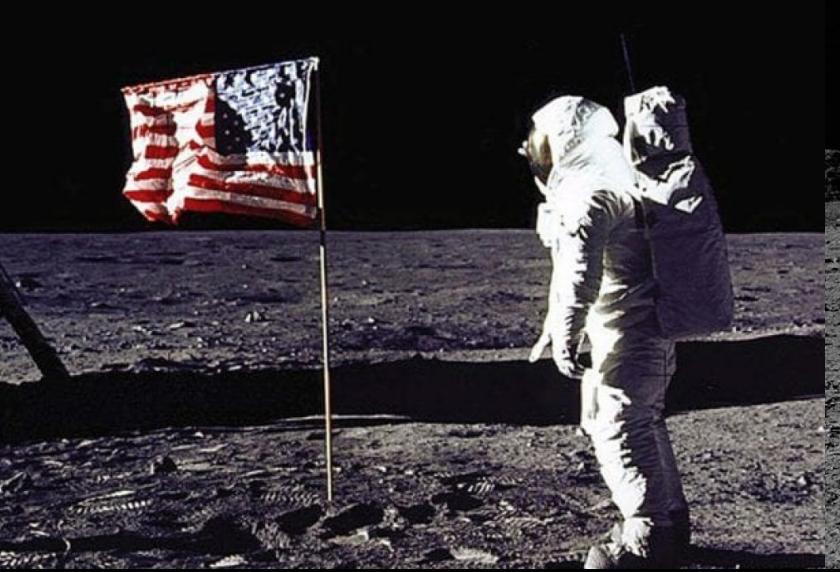
---

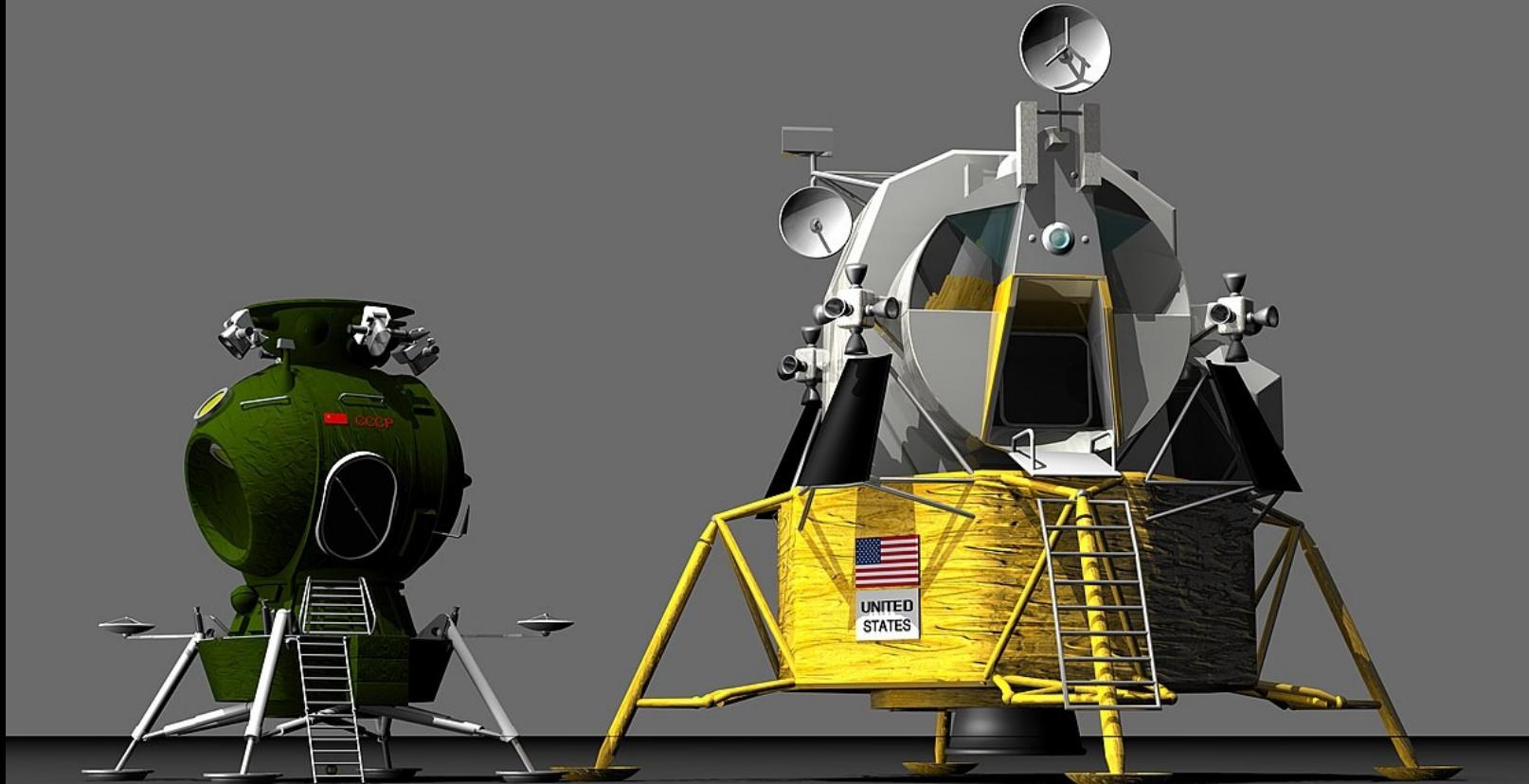
# Sancus publication track



- Overall: 29 papers (6 A\*, 6 A, 5 journal, 7 workshop, 5 others)
- Do invest in systems **foundation**, don't blind stare on A\* ...









## **Key #8: Re-invent yourself**

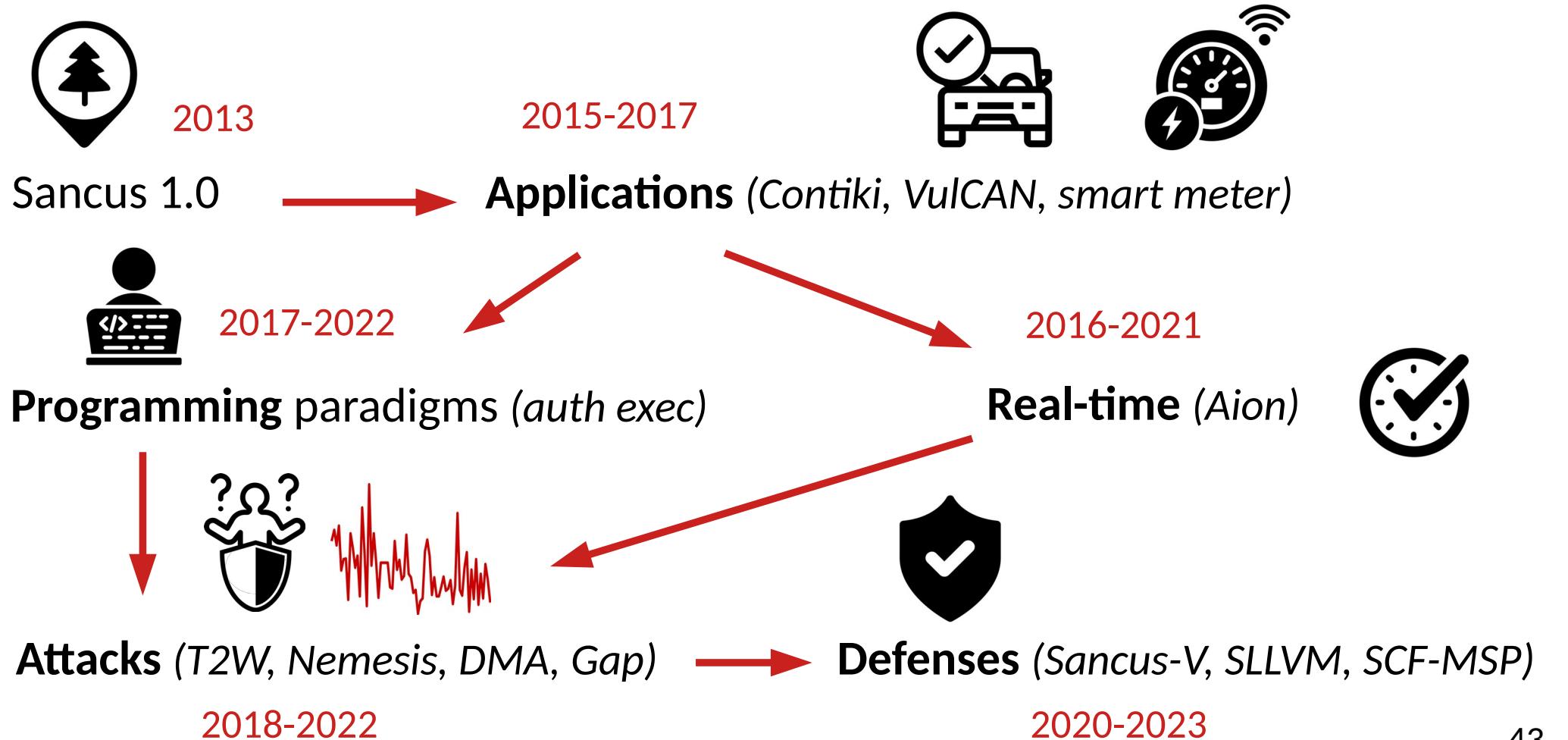
---



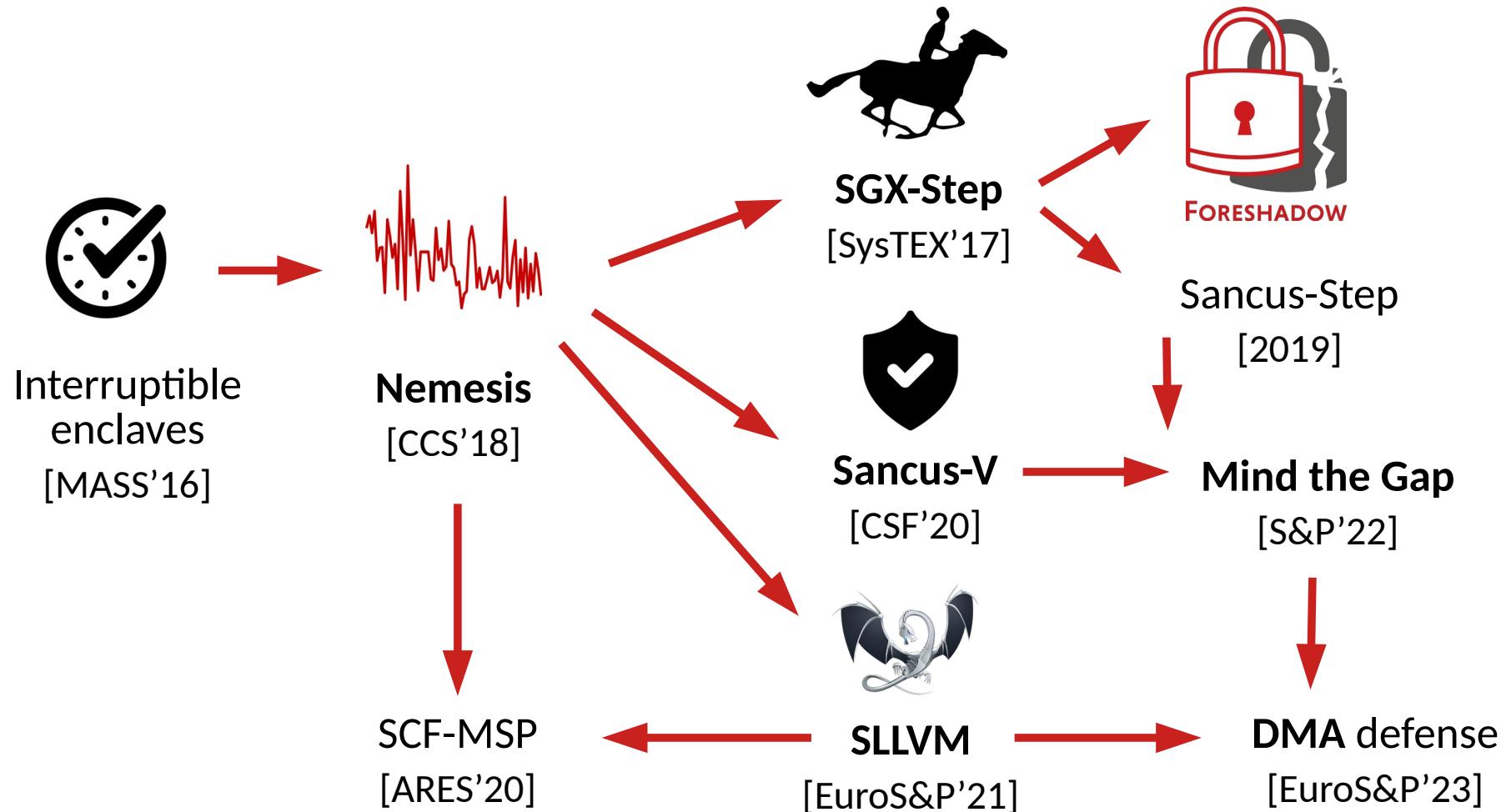
PIVOT!

PIVOT!

# Sancus is dead, long live Sancus!



# Sancus attack research: The gift that keeps giving



# Conclusion: Sancus's 7 magic ingredients

- 1) Research team
- 2) Relevant niche
- 3) Open source
- 4) Usable systems
- 5) Science communication
- 6) When to publish
- 7) Pivot

