Security by Design

+ some bonus material at the end

Erik Poll

Digital Security group

Radboud University Nijmegen

INTERSCT.

Thanks to NWO and Min. J&V for funding

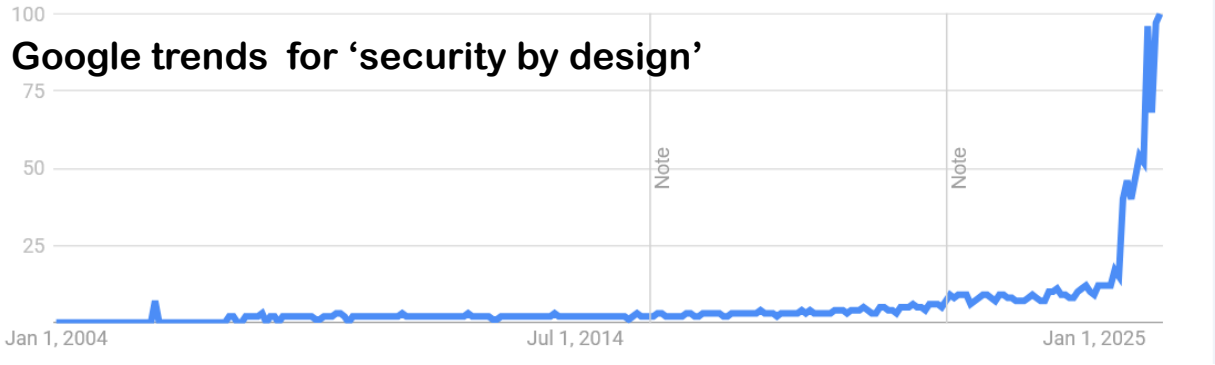

SECURITY
by
DESIGN


SECURITY
BY
DESIGN

narrow vs **broad** definition

Common misunderstandings

- ‘Security by Design’ is a new thing



- ‘Security by Design’ is something you *only* do in the early design phase

- ‘Security by Design’ is going to solve all your security problems

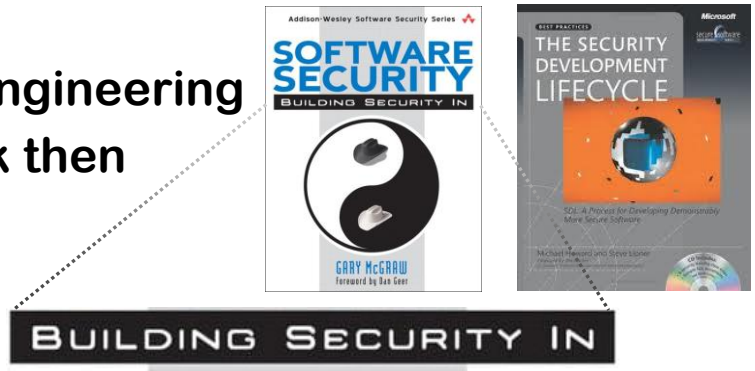
There are no silver bullets in security

Security by Design

Security by Design

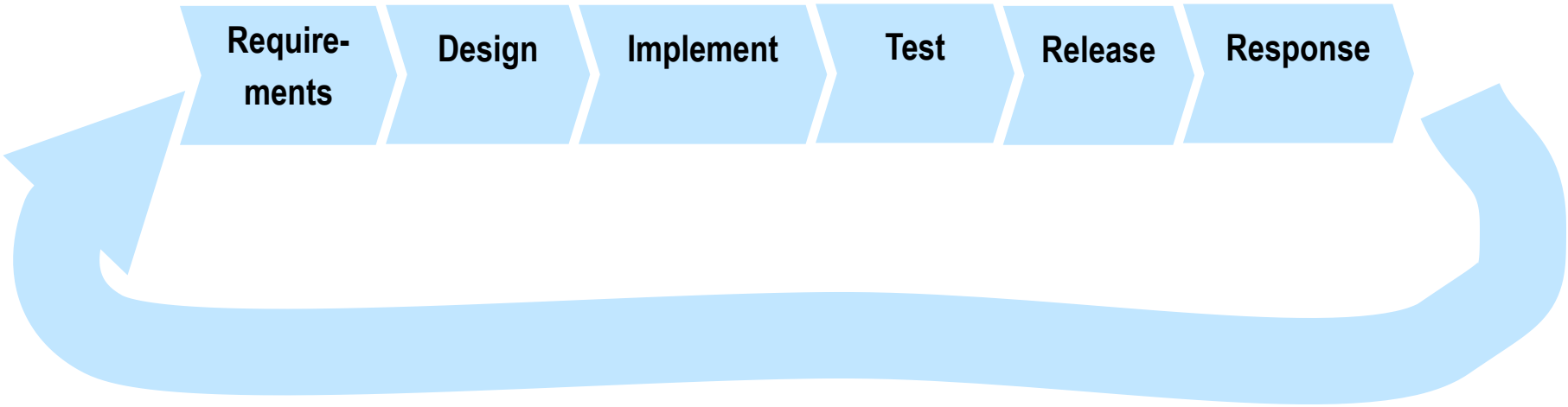
Not one activity in early design phase,
but *range* of activities *throughout* the entire development lifecycle

- **Introduced in early 2000s in software engineering but not called 'Security by Design' back then**



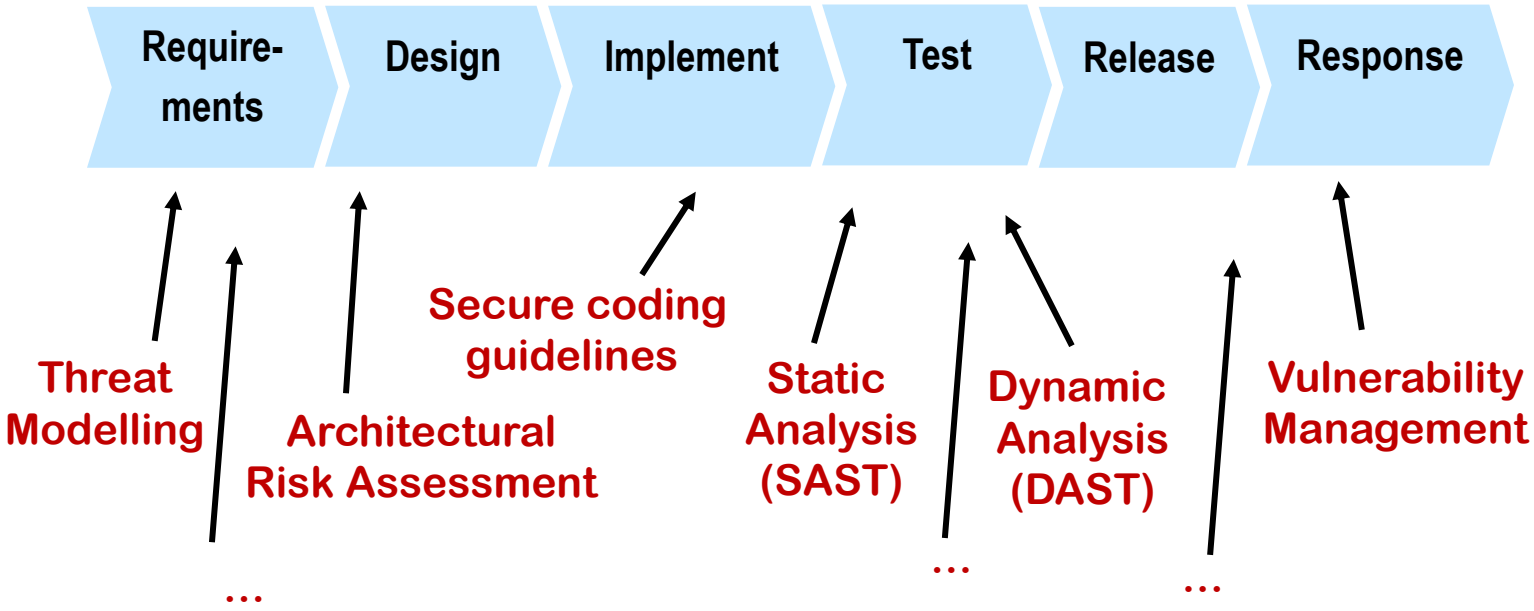
- ***Many* methodologies, standards, frameworks, … since then, with some additions and tweaks, but same overall idea**

Software development lifecycle



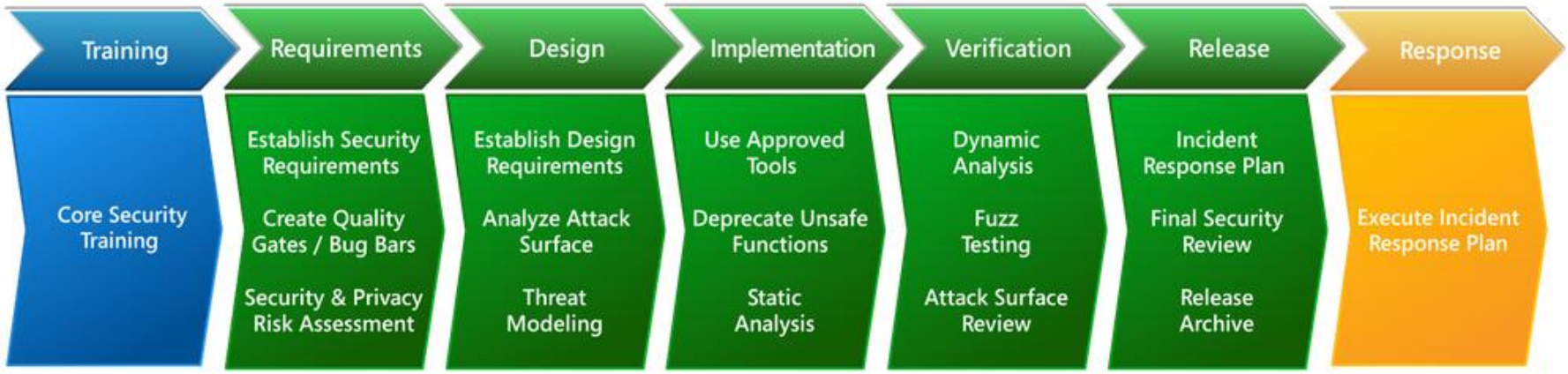
Security by Design

Security activities at different points in the lifecycle, for instance

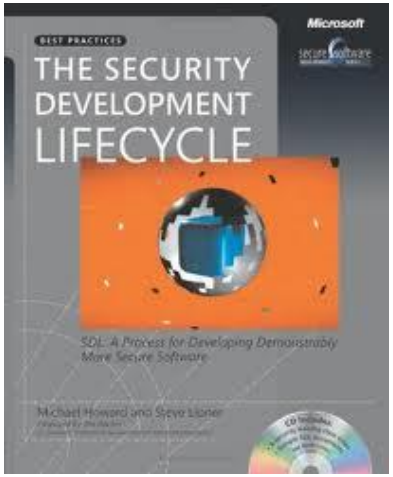


Many standards,
all describing many similar activities,
at different levels of granularity & grouped in different ways

Microsoft SDL (2012 version)



5x3 activities in lifecycle
plus training & incidence response



Security by Design as defined in EN 18031

EN 18031, the standard for the EU Radio Equipment Directive (RED), refers to six other standards for the definition

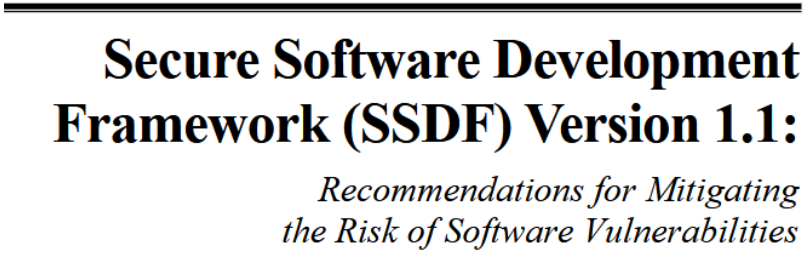
A.2.2 Security by design

Effective security management requires established security by design processes, which is not covered by this document which defines common security requirements for the equipment. Examples of security by design process standards which would aid in the ability to satisfy the security requirements include:

- IEC 62443-4-1[1]: Security for industrial automation and control systems – Part 4-1: Secure product development lifecycle requirements
- NIST 800-160[16]: Systems Security Engineering; Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems
- NIST 800-218[17]: Secure Software Development Framework (SSDF)
- Microsoft Security Development Lifecycle (SDL)
- SAFECode Fundamental Practices for Secure Software Development
- GSMA FS.16 NESAS Development and Lifecycle Security Requirements

NIST SSDF

NIST Secure Software Development Framework (2022)
lists 20 practices in four groups
with a further breakdown into 43 actions


NIST Special Publication 800-218

Secure Software Development Framework (SSDF) Version 1.1:

Recommendations for Mitigating the Risk of Software Vulnerabilities

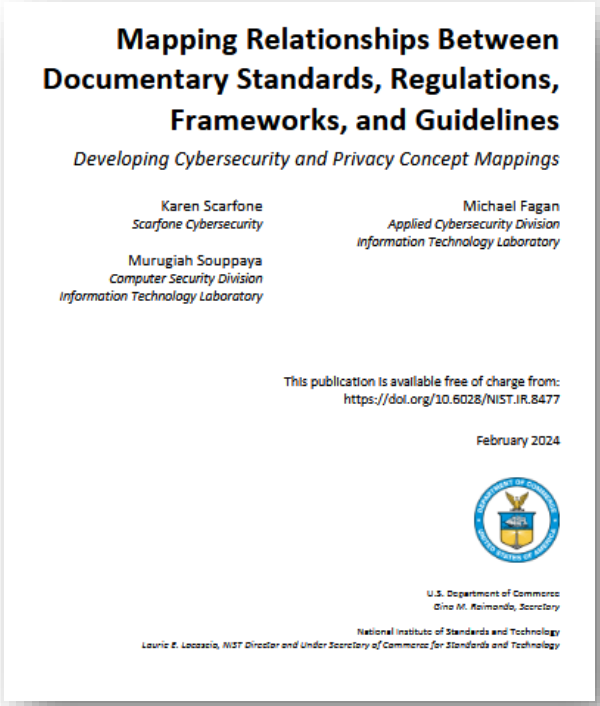
Practices & actions taken from 25 earlier standards, incl.

Microsoft SDL, BSIMM12, ISA/IEC 62443, IDA SOAR, CNCF FSSCP,
OWASP SAMM, OWASP ASVS, OWAPS SCVS, PCI SSLC,
NIST IR8397, NIST SP800-52, SP 800-160, SP 800-161,
NIST CSF, NIST LAB,
SafeCode Fundamental Practices For Secure Software Development,
SafeCode SIC, SafeCode TPC,
BSA Framework for Secure Software, EO 14028, …

How to cope with ever more security standards?

- **OWASP OpenCRE initiative provides mappings between security requirements [https://www.opencre.org]**

- **NIST released a methodology for mapping relations between cybersecurity standards (IR 8477, 2024)**



Mapping Relationships Between Documentary Standards, Regulations, Frameworks, and Guidelines

Developing Cybersecurity and Privacy Concept Mappings

Karen Scarfone
Scarfone Cybersecurity

Michael Fagan
Applied Cybersecurity Division
Information Technology Laboratory

Murugiah Souppaya
Computer Security Division
Information Technology Laboratory

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8477

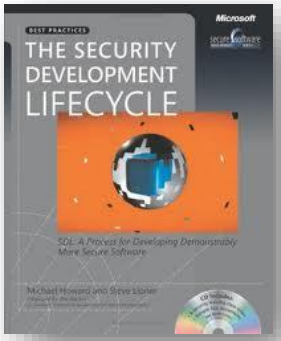
February 2024

U.S. Department of Commerce
Gina M. Raimondo, Secretary

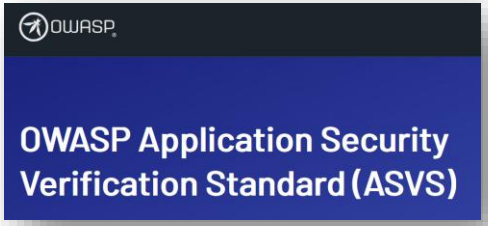
National Institute of Standards and Technology
Laurie E. Locascio, NIST Director and Under Secretary of Commerce for Standards and Technology

How to cope with ever more security standards?

Some security advice is about the process



Other security advice is about the product



Some standards include guidance for both aspects

TNO Product Security Innovation Radar

Francesco Pizzocolo & colleagues at TNO created a draft taxonomy for product cybersecurity

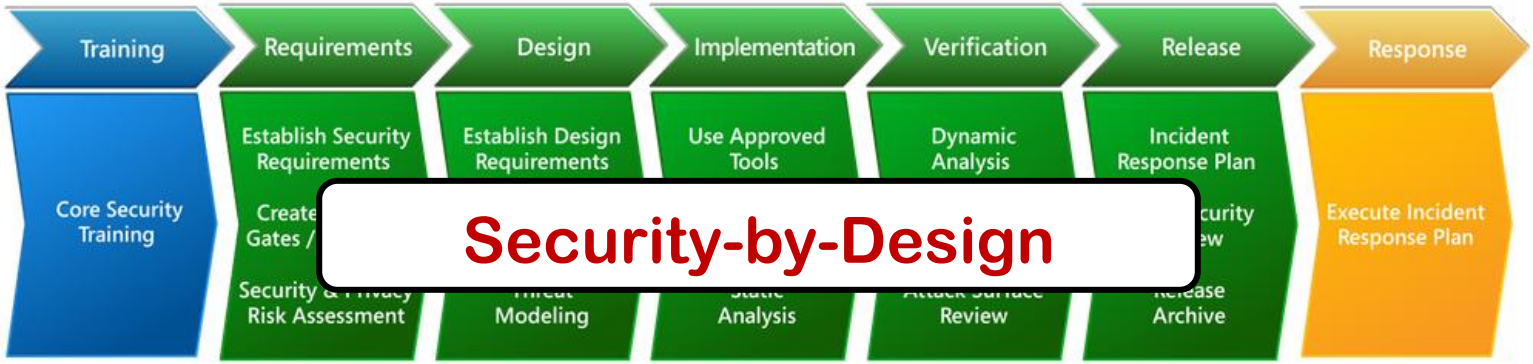
> A working taxonomy of product cybersecurity for software-intensive, high-tech and dual-use systems. Five clusters, thirty-seven subclusters covering design, development, supply chain, governance and post-deployment.

https://bhoc-radar.cst-lab.nl

that they would like your feedback on

What's changed in software security in the past 20 years?

What's changed? New *slogans*



Shifting Left

Thinking about security earlier

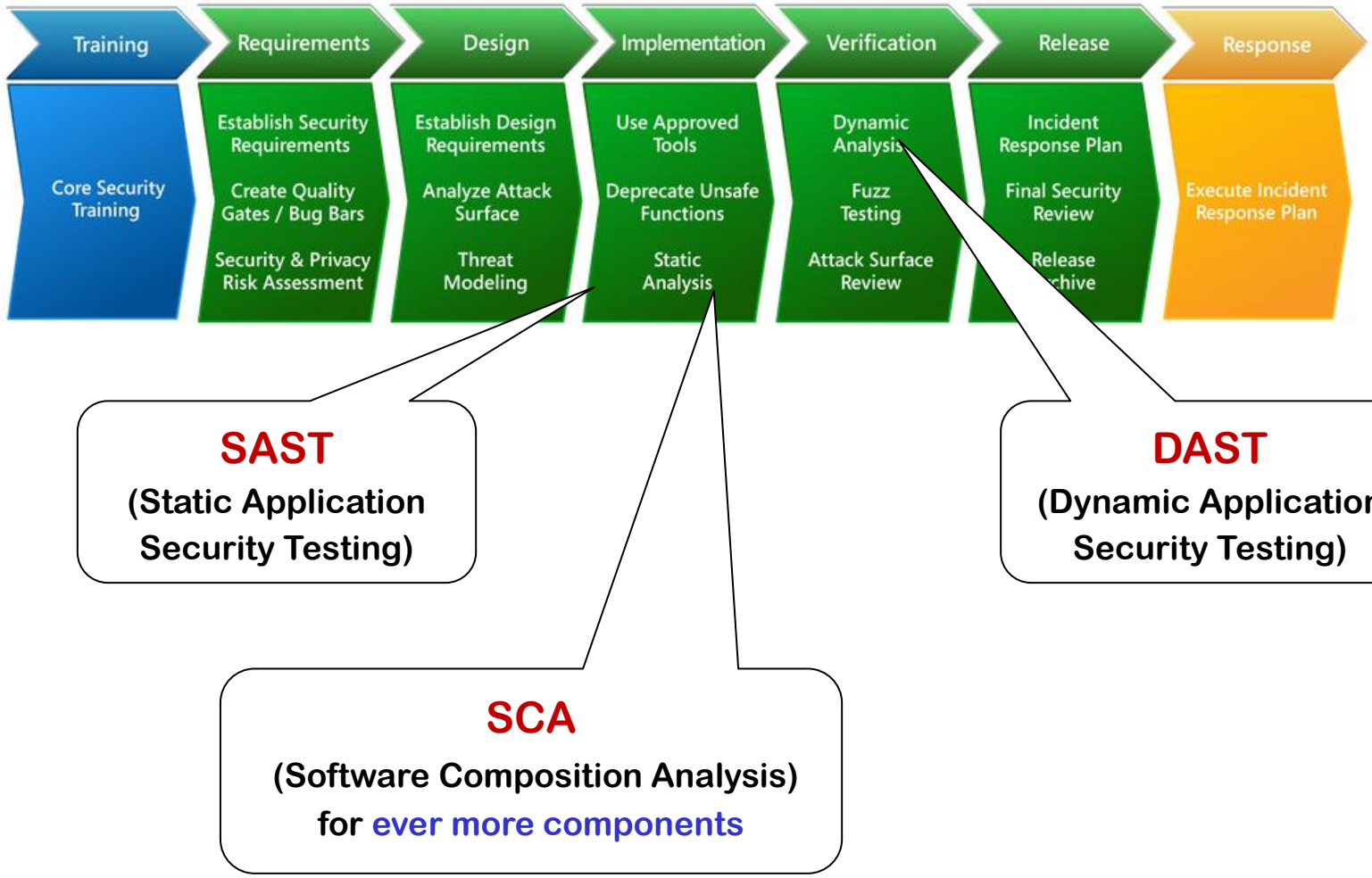
Shifting Right

Accepting & preparing for security failures

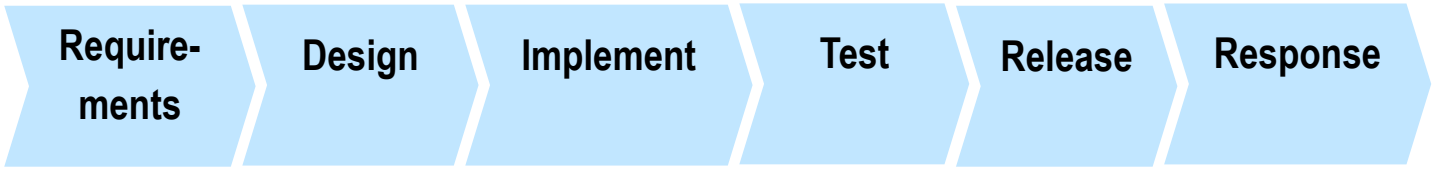
Shifting Down

Tacklng security lower in tech stack:

What's changed? More *acronyms*



Shifting Left



0. Wait for problems

1. Hire a pen-tester

2. Run DAST tool, e.g. fuzzer

3. Run SAST tool, e.g. SonarCube, Fortity, CodeQL, …

4. Choose Safer APIs, e.g. PreparedStatemet

5. Choose safer programming language,
e.g. TypeScript or Rust

6. Threat Modelling
and architectural risk analysis of attack surfaces

Shifting Down

Best way to shift left:

shift *down*

by addressing security lower down in technology stack

For instance, using

- safer APIs, less prone to injection attacks
- safer programming language, e.g. TypeScript & Rust
- built-in update mechanisms in the operating system, eg iOS & Android
- built-in authentication mechanism, e.g. biometrics on mobile
- …

But also: Shifting Right

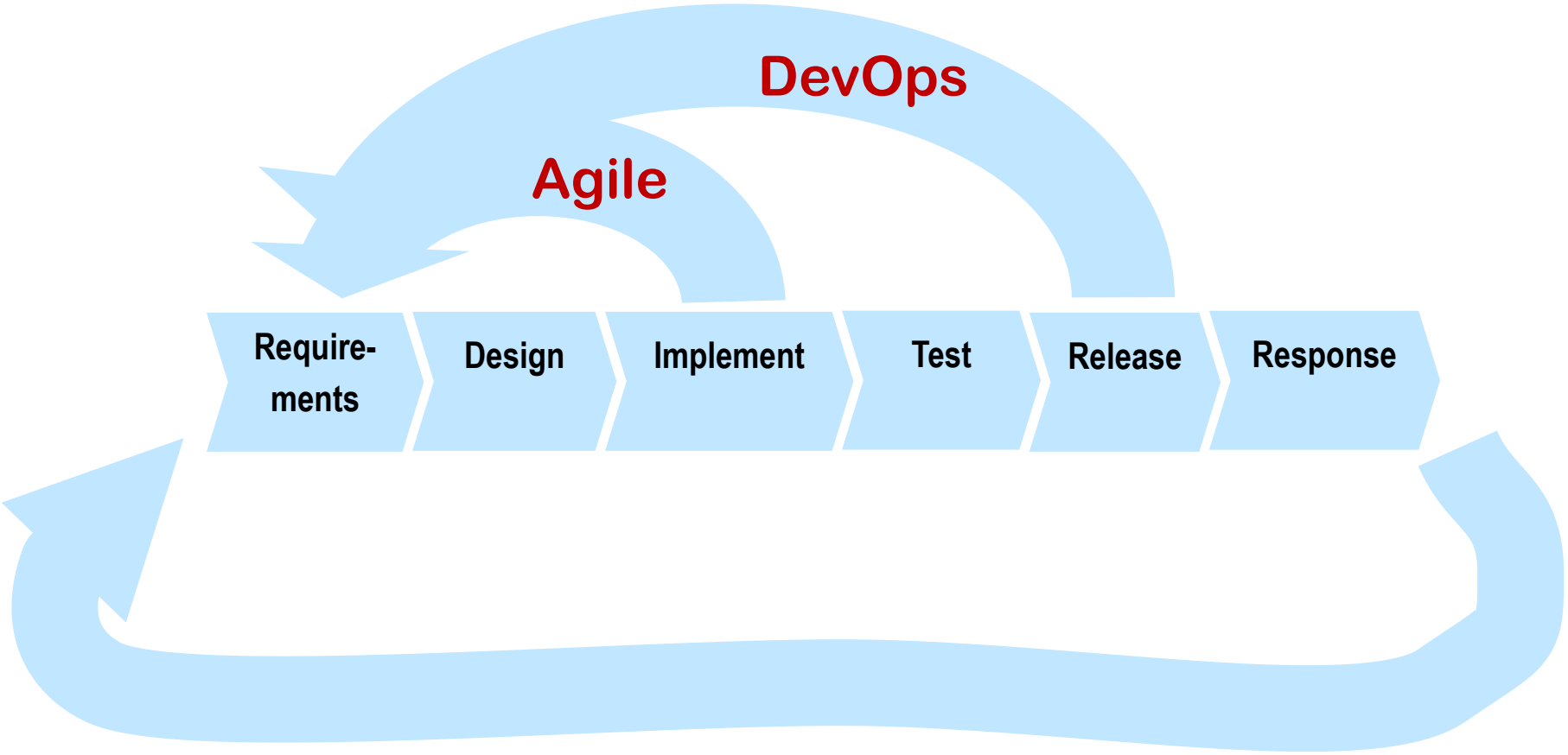
Fallacy to think that all security problems can be prevented!

You have to prepare for & cope with failure

- monitoring & response, ie. detection instead of prevention
  - less for software developer and more/also for operator?
- be prepared to patch & update
- vulnerability management
- …

What's changed in *software engineering*
in the past 20 years?

1. Software development even more iterative



1. Agile & DevOps

Early security-by-design methodologies used waterfall model as frame of reference

How can we cope with Agile or DevOps?

We cannot do pen-test for every new feature or weekly release

No new activities, but changes in when & how often to do them

And: *more important to shift left!* Eg.

- use DAST and – further to the left – SAST
- train developers
- integrate SAST & DAST into CD/CI pipelines
- …

The term DevSecOps was introduced as confusing new buzzword

2. Code repositories

Lots of code reuse from code repositories

github, Maven, PyPi, ...

New attack vector: supply chain attacks

Eg Log4J, SolarWinds, XZ utils

New countermeasures

1) SCA (Software Composition Analysis)
   static analysis tools to check software supply chain for CVEs
2) SBOM (Software Bill of Materials)

And more standards: e.g. OWASP SCVS

3. 'Services'

Software increasingly built using (cloud-based) *services*
instead of libraries as *components*
with SaaS, Service-Oriented Architectures, micro-services, cloud APIs

Incl. services for the software build process itself

This introduces

- more attack surface
- need for authentication to cloud APIs

New security risk: leaking credentials (JWT tokens, AWS security tokens, ...)

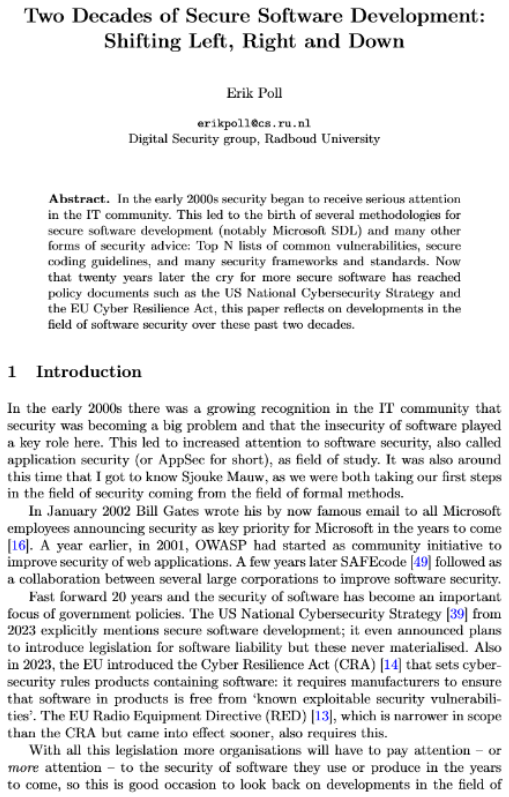
New countermeasures:

1) SAST tools for secret scanning, eg TruffleHog
2) proposals for SaaSBOMs

More details?

**“Two decades of secure software developmen
Shifting left, right and down”**

In: Juggling Formal Methods and Security
LNCS volume 16365,
DOI:10.1007/978-3-032-20684-8_12
Springer Nature, 2026



Two Decades of Secure Software Development:
Shifting Left, Right and Down

Erik Poll

erikpoll@cs.ru.nl
Digital Security group, Radboud University

Abstract. In the early 2000s security began to receive serious attention in the IT community. This led to the birth of several methodologies for secure software development (notably Microsoft SDL) and many other forms of security advice: Top N lists of common vulnerabilities, secure coding guidelines, and many security frameworks and standards. Now that twenty years later the cry for more secure software has reached policy documents such as the US National Cybersecurity Strategy and the EU Cyber Resilience Act, this paper reflects on developments in the field of software security over these past two decades.

1 Introduction

In the early 2000s there was a growing recognition in the IT community that security was becoming a big problem and that the insecurity of software played a key role here. This led to increased attention to software security, also called application security (or AppSec for short), as field of study. It was also around this time that I got to know Sjouke Mauw, as we were both taking our first steps in the field of security coming from the field of formal methods.

In January 2002 Bill Gates wrote his by now famous email to all Microsoft employees announcing security as key priority for Microsoft in the years to come [16]. A year earlier, in 2001, OWASP had started as community initiative to improve security of web applications. A few years later SAFEcode [49] followed as a collaboration between several large corporations to improve software security.

Fast forward 20 years and the security of software has become an important focus of government policies. The US National Cybersecurity Strategy [39] from 2023 explicitly mentions secure software development; it even announced plans to introduce legislation for software liability but these never materialised. Also in 2023, the EU introduced the Cyber Resilience Act (CRA) [14] that sets cybersecurity rules products containing software: it requires manufacturers to ensure that software in products is free from ‘known exploitable security vulnerabilities’. The EU Radio Equipment Directive (RED) [13], which is narrower in scope than the CRA but came into effect sooner, also requires this.

With all this legislation more organisations will have to pay attention – or *more* attention – to the security of software they use or produce in the years to come, so this is good occasion to look back on developments in the field of

What's changed in security in the past few years?

More legislation, incl. Cyber Resilience Act (CRA)

Regulation for 'products with digital elements'

- **Broader in scope than Radio Equipment Directive (RED)**
- **Complements NIS2 for 'services'**

CRA requires products placed on market

with "no known exploitable vulnerabilities"

**"designed, developed and produced …
to ensure an appropriate level of cybersecurity "**

and that

**"manufacturers take security seriously
throughout a product's lifecycle"**

The CRA & RED come with yet more associated standards, EN 40000 & EN 18031

[Banksy, 2006]

What about LLMs?

Anthropic's Claude Mythos Finds Thousands of Zero-Day Flaws Across Major Systems

Ravie Lakshmanan Apr 08, 2026

CNN

Anthropic suspends all access to Mythos model after US government bans foreign nationals use

RTL Z

Waarom Europa zich zorgen maakt over machtsgreep VS bij AI-koploper Anthropic

Security & AI

Questions about security & AI are much broader than just using LLMs for bug finding:

- **Security *of* AI**
  - **using AI in your software application**
  - **more generally, using AI in your processes**
  - **using AI to generate code**
- **Security *with* AI, or AI *for* security**
  - **using AI to secure systems, e.g. anomaly detection in your SOC**
  - **using AI to find security flaws in software**

Impact of LMM on software security

- How good, secure, maintainable will AI-generated code be?

- Will LLMs be better at
  - finding flaws,

    with an acceptable rate of false positives!
  - writing patches,
  - writing exploits?

- How good will they be at writing an exploit given a patch?

Bugcopalypse?

- In long run, tools to find bugs are only good
- In short run, the rising number of discovered bugs is a challenge for software makers
  - And users of software can expect more patches…

- Changes in attacker behaviour are more difficult to predict

  For attackers, AI may be more valuable for scaling up attacks or exploiting known vulnerabilities than for finding new ones
  (E.g. recall the unexpectedly low impact of Log4J)

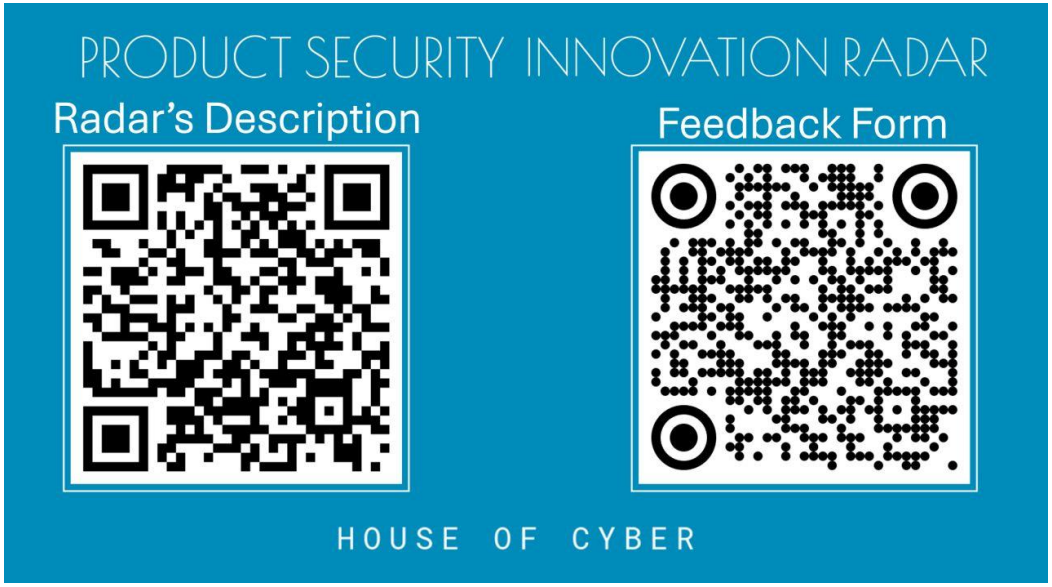
Wrapping up

Conclusions

- Security by Design is nothing new and not only about the design phase
  - Analogy with Privacy by Design is misleading!
- Many, very similar standards on how to do this
  - Often not using the term security-by-design.
  - The problem is not that we do not know what to do, but that there are so many things we can do.
- Just pick one standard, see what works for you, and try to shift left
  - but don't forget to also shift right
  - and be aware that the best way to shift left is to shift down
- Good that the CRA forces more companies to take software security seriously, but forced compliance can foster cover-your-ass security
  - Will this really fix market failures that are common in security?
- It will be interesting to see the impact of LMMs on software development and software security – and on security in general…

TNO Product Security Innovation Radar

Strong opinions on what is needed or missing?



https://bhoc-radar.cst-lab.nl

https://form.typeform.com/to/dI8IpNc0