You’re not secure by design if you’re not memory safe!

ONE Conference 2025

Hugo van de Pol, Marc Schoolderman

About us

Marc Schoolderman

- Systems software engineer at Tweede Golf
- Former researcher and computer science teacher at Radboud University Nijmegen, The Netherlands

Hugo van de Pol

- Director at Tweede Golf since 2018
- Board member at Trifecta Tech Foundation
- Adoption of memory safe technology (Rust)

Confessions of a car person

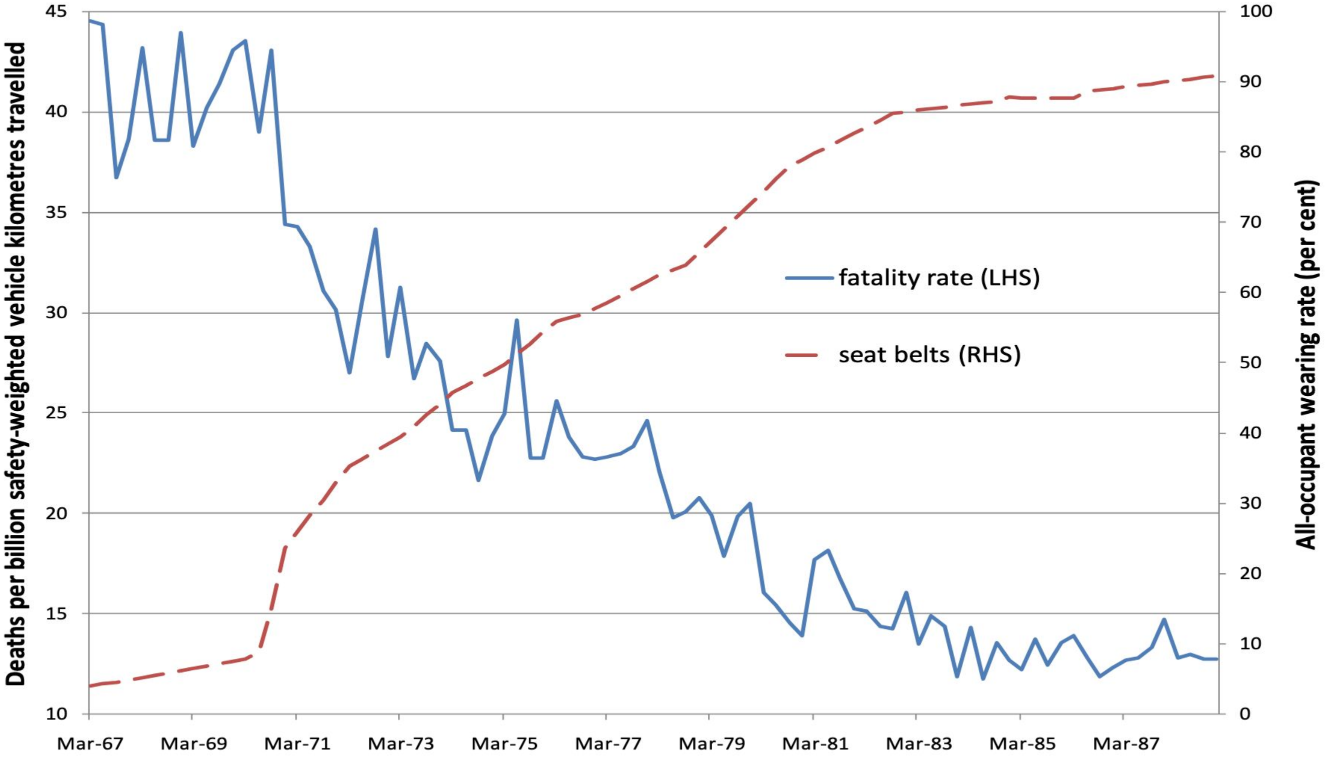
They look cool!

Freedom for the individual!

Economic benefits!

Driving ~~is dangerous~~ *exciting*!

>70% of car deaths in 1970: preventable!



tweede golf

Source: Bureau of Infrastructure, Transport and Regional Economics, Canberra, Australia (2009)

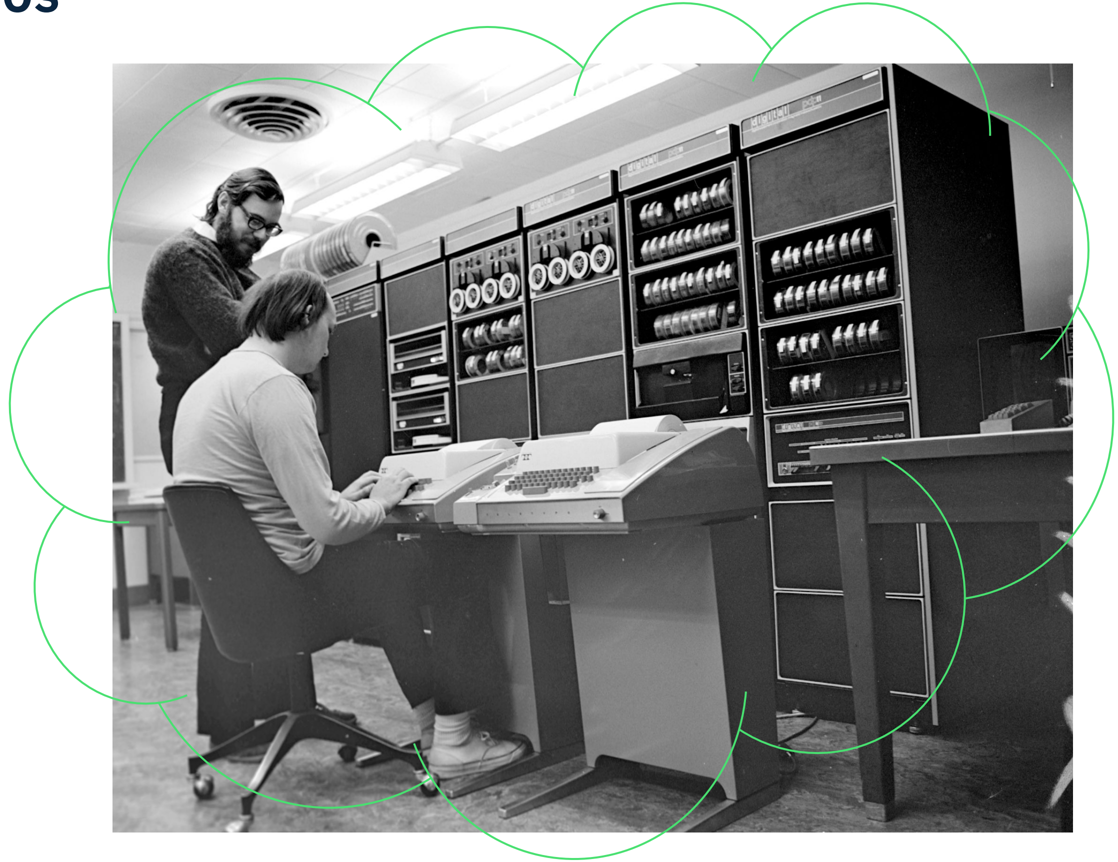
The Era of Computing: 1970s

Software can do amazing things!

More control over your life!

Economic benefits!

Code is ~~dangerous~~ *exciting*!



Pioneers of modern systems programming Dennis Ritchie & Ken Thompson, in 1972

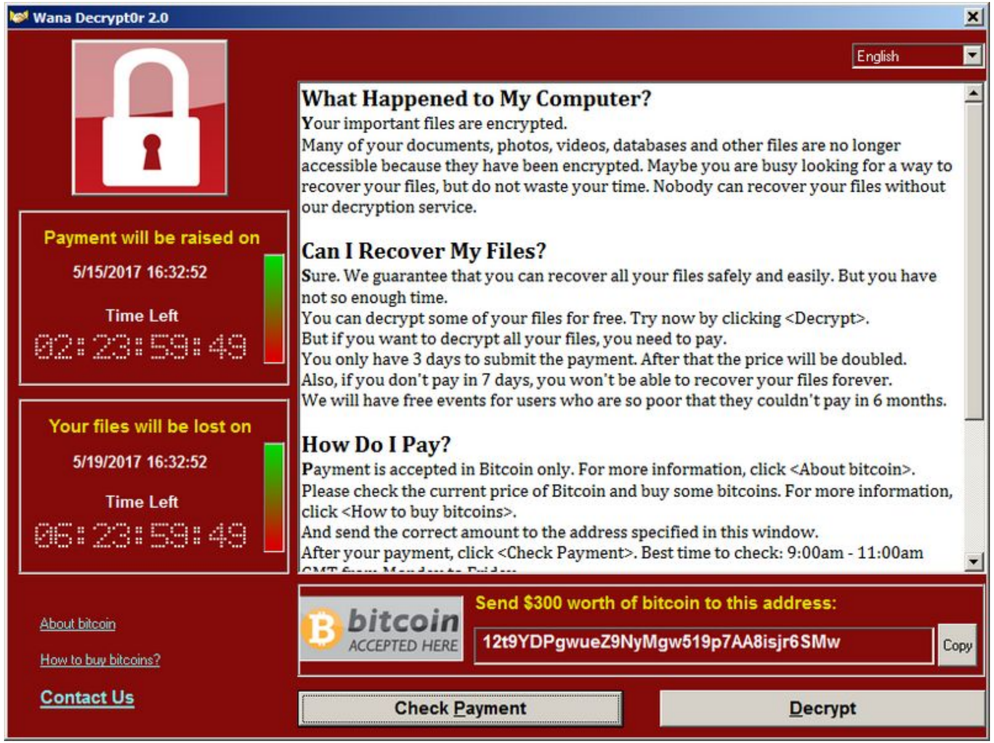
Fast forward to the 2020s

Software can do amazing things!

More control over your life!

Economic benefits!

Code is ~~dangerous~~ ~~*exciting*~~ *dangerous*!



Payment will be raised on

5/15/2017 16:32:52

Time Left

02:23:59:49

Your files will be lost on

5/19/2017 16:32:52

Time Left

06:23:59:49

What Happened to My Computer?

Your important files are encrypted.

Many of your documents, photos, videos, databases and other files are no longer accessible because they have been encrypted. Maybe you are busy looking for a way to recover your files, but do not waste your time. Nobody can recover your files without our decryption service.

Can I Recover My Files?

Sure. We guarantee that you can recover all your files safely and easily. But you have not so enough time.

You can decrypt some of your files for free. Try now by clicking <Decrypt>.

But if you want to decrypt all your files, you need to pay.

You only have 3 days to submit the payment. After that the price will be doubled.

Also, if you don't pay in 7 days, you won't be able to recover your files forever.

We will have free events for users who are so poor that they couldn't pay in 6 months.

How Do I Pay?

Payment is accepted in Bitcoin only. For more information, click <About bitcoin>.

Please check the current price of Bitcoin and buy some bitcoins. For more information, click <How to buy bitcoins>.

And send the correct amount to the address specified in this window.

After your payment, click <Check Payment>. Best time to check: 9:00am - 11:00am GMT from Monday to Friday.

Send $300 worth of bitcoin to this address:

12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw

Key motivation 1/4

1. **Digital threats will continue to rise in scale & sophistication.**

ENISA THREAT LANDSCAPE 2024

July 2023 to June 2024

SEPTEMBER 2024

State actors, cybercriminals, “hacktivists”

July 2022 - July 2023: **~24k** reported vulnerabilities

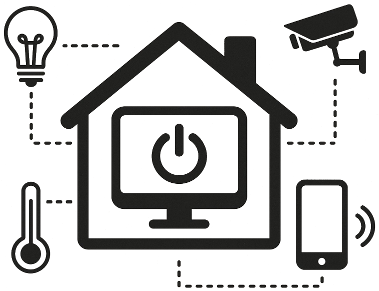
July 2023 - July 2024: **~33k** reported vulnerabilities

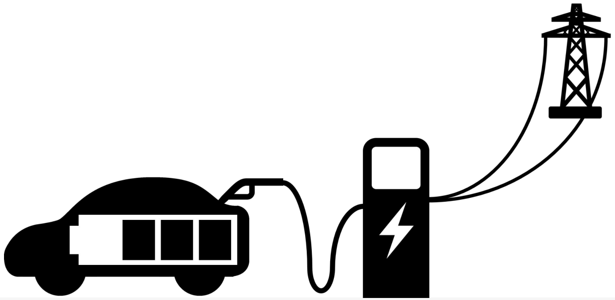
- **37% increase**

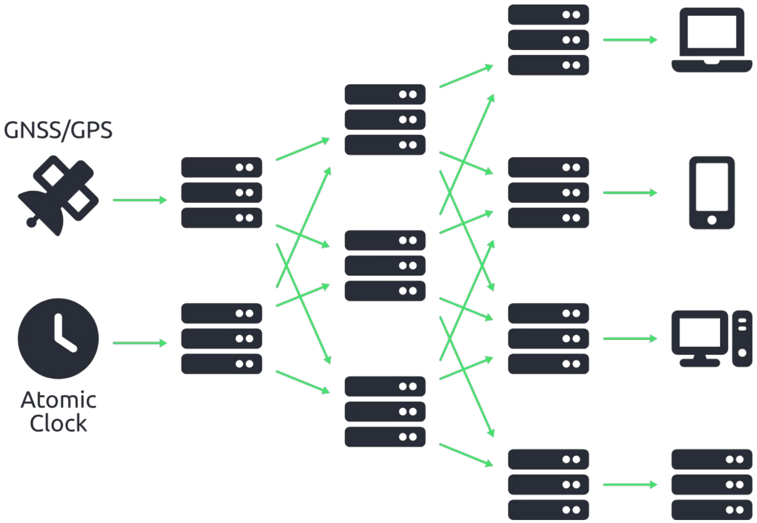
Enisa Threat Landscape 2024

Key motivation 2/4

2. **Our reliance on digital technologies will not decrease.**







Key motivation 3/4

3. **Costs of after-the-fact patching are becoming unsustainable.**

Security Navigator 2025

Research driven insights to build a safer digital society

“We […] are still struggling to stem the flood”

“We cannot patch fast enough”

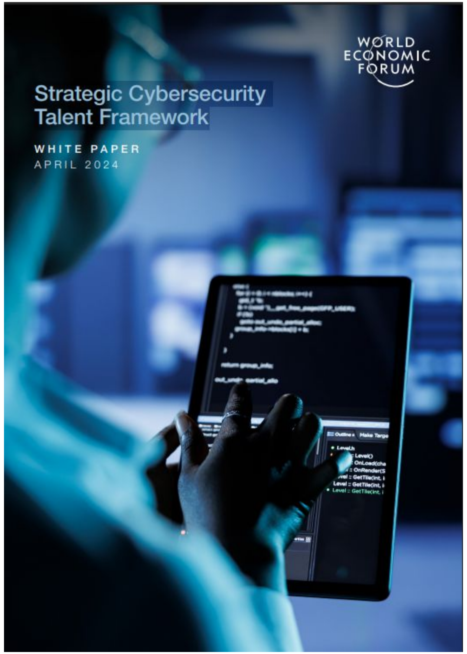
Orange Cyberdefense Security Navigator 2025

Key motivation 4/4

4. **Shortage of cybersecurity professionals makes a reactive approach untenable.**

“Today [2024], **four million** workers in the cybersecurity industry are needed worldwide”

[WEF Strategic Cybersecurity Talent Framework White Paper]



WORLD ECONOMIC FORUM

Strategic Cybersecurity Talent Framework

WHITE PAPER
APRIL 2024

We need a different approach

We need more

Security by Design

Security by Design (SbD): quick recap

- Fundamental concept in important EU legislation such as the **Cyber Resilience Act (CRA)**
- **Annex I: Essential Requirements**, including:
  - Risk-based approach to cyber security
  - … without known exploitable vulnerabilities…
  - Limit attack surface
  - Reduce the impact of incidents
- **SbD at all layers, down to our very building blocks**

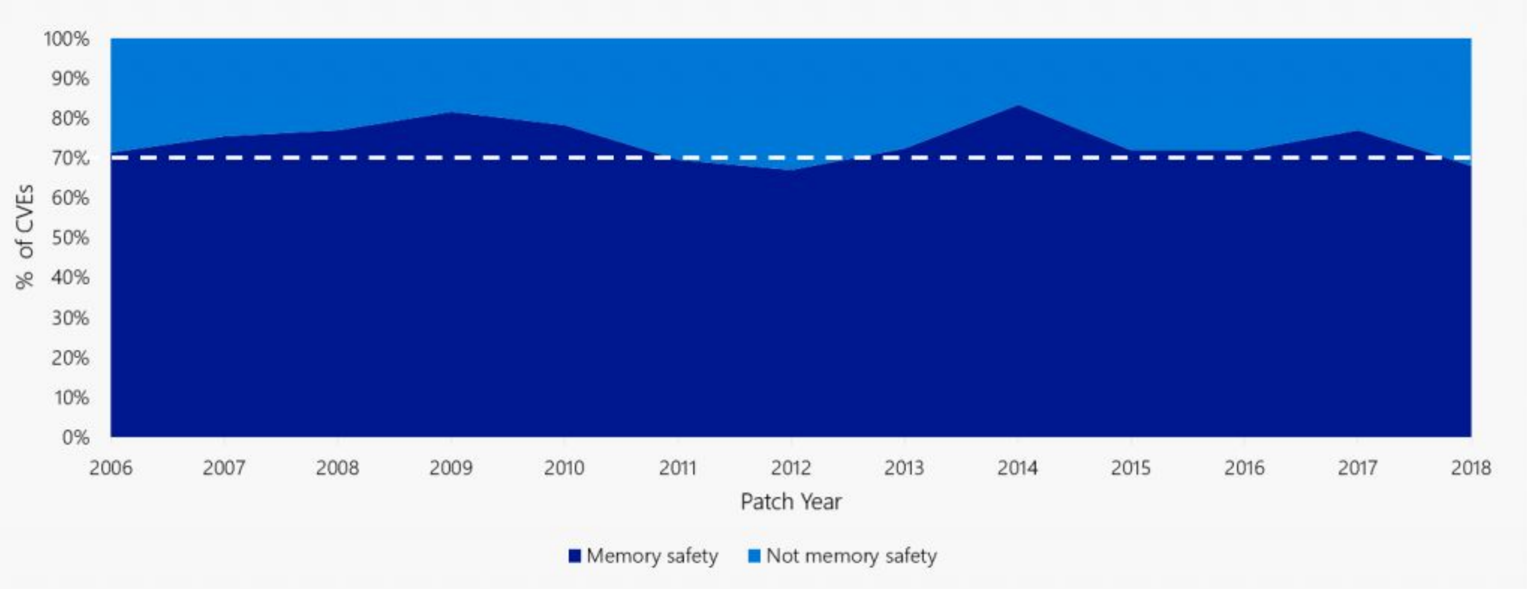
We need better building blocks

We need building blocks that are

Memory safe

A large class of vulnerabilities can be avoided

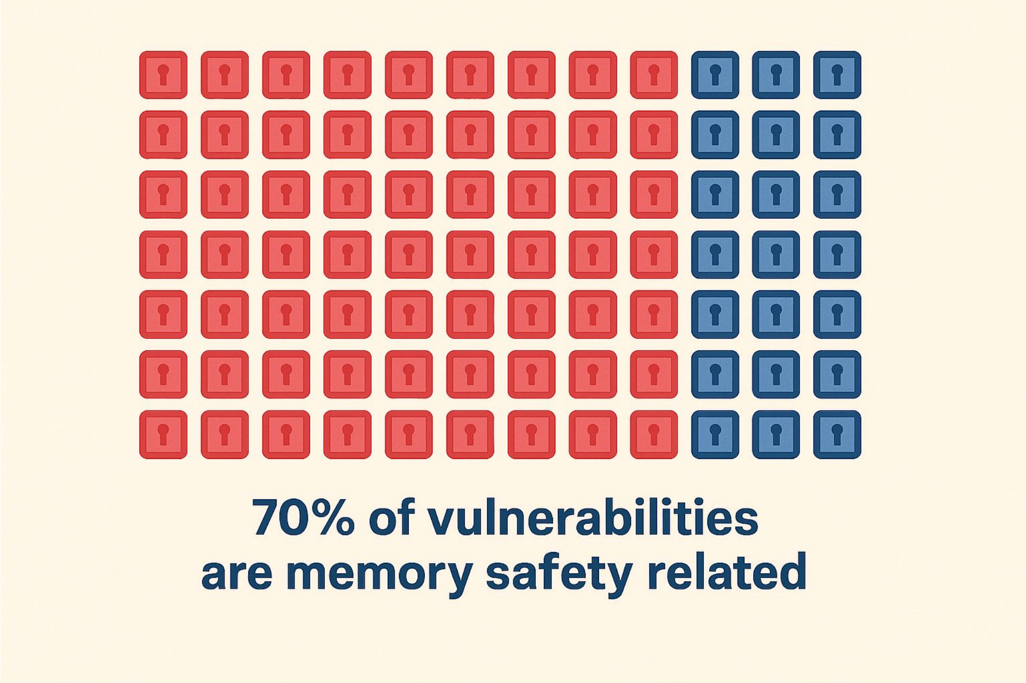
~Up to 70% of vulnerabilities in **memory unsafe** code bases is memory safety related!



tweede golf

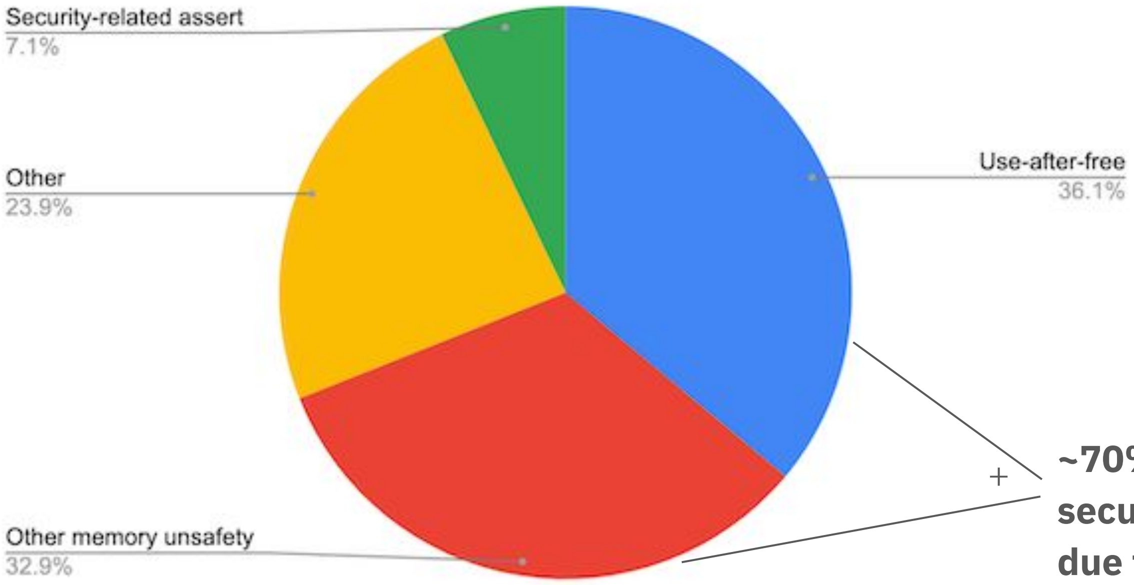
Source: Microsoft ‘We need a safer systems programming language‘, July 2019

There’s a lot to gain!



70% of vulnerabilities are memory safety related

Similar results in Chromium



~70% of critical severity security bugs in Chromium are due to memory safety

Chromium Security / Memory safety

0-day exploits tracked by Google Project Zero

CVE-2025-24085	Apple	iOS	Memory Corruption	Use after free in CoreMedia
CVE-2025-24200	Apple	iOS	Security Feature Bypass	A physical attack may disable USB Restricted Mode
CVE-2025-21391	Microsoft	Windows	Logic Error	Windows Storage Elevation of Privilege Vulnerability
CVE-2025-21418	Microsoft	Windows	Memory Corruption	Windows Ancillary Function Driver for WinSock Elevation of Privilege Vulnerability
CVE-2025-24201	Apple	WebKit	Memory Corruption	OOB write
CVE-2025-26633	Microsoft	Windows	Security Feature Bypass	Microsoft Management Console Security Feature Bypass
CVE-2025-24993	Microsoft	Windows	Memory Corruption	Windows NTFS Remote Code Execution Vulnerability
CVE-2025-24985	Microsoft	Windows	Memory Corruption	Windows Fast FAT File System Driver Remote Code Execution Vulnerability
CVE-2025-24983	Microsoft	Windows	Memory Corruption	Windows Win32 Kernel Subsystem Elevation of Privilege Vulnerability
CVE-2025-24984	Microsoft	Windows	Information Disclosure	Windows NTFS Information Disclosure Vulnerability

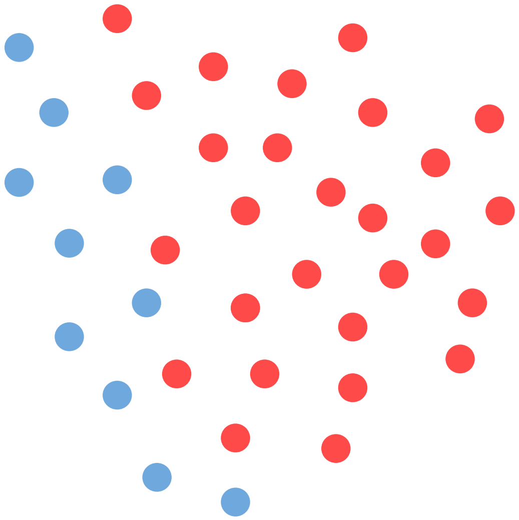
0-day exploits tracked by Google Project Zero

CVE-2025-24991	Microsoft	Windows	Information Disclosure	Windows NTFS Information Disclosure Vulnerability
CVE-2025-22225	VMWare	VMware ESXi	Memory Corruption	OOB VMware ESXi
CVE-2025-27363	FreeType	FreeType	Memory Corruption	OOB write
CVE-2025-2783	Google	Chrome	Logic Error	Windows Chrome sandbox escape
CVE-2025-31200	Apple	iOS	Memory Corruption	Memory Corruption in CoreAudio
CVE-2025-31201	Apple	iOS	PAC bypass	Arbitrary read and write
CVE-2025-29824	Microsoft	Windows	Memory Corruption	Windows Common Log File System Driver Elevation of Privilege Vulnerability
CVE-2025-32701	Microsoft	Windows	Memory Corruption	Windows Common Log File System Driver Elevation of Privilege Vulnerability
CVE-2025-30397	Microsoft	Windows	Memory Corruption	Scripting Engine Memory Corruption Vulnerability
CVE-2025-30400	Microsoft	Windows	Memory Corruption	Microsoft DWM Core Library Elevation of Privilege Vulnerability
CVE-2025-32709	Microsoft	Windows	Memory Corruption	Windows Ancillary Function Driver for WinSock Elevation of Privilege Vulnerability
CVE-2025-32706	Microsoft	Windows	Memory Corruption	Windows Common Log File System Driver Elevation of Privilege Vulnerability

0-day exploits tracked by Google Project Zero

CVE-2025-5419	Google	Chrome	Memory corruption	OOB write in V8
CVE-2025-21479	Qualcomm	GPU	Memory corruption	Arbitrary physical write vulnerability
CVE-2025-21480	Qualcomm	GPU	Memory corruption	Arbitrary physical write vulnerability
CVE-2025-27038	Qualcomm	GPU	Memory corruption	UAF
CVE-2025-6554	Google	Chrome	Memory corruption	Type confusion in V8
CVE-2025-33053	Microsoft	Windows	Logic Error	Internet Shortcut Files Remote Code Execution Vulnerability
CVE-2025-6558	Google	Chrome	Memory corruption	Insufficient validation of untrusted input
CVE-2025-53770	Microsoft	Sharepoint	Logic Error	Deserialization of untrusted data
CVE-2025-43300	Apple	iOS	Memory corruption	Memory corruption in ImageIO
CVE-2025-21043	Samsung	Samsung Mobile	Memory corruption	OOB write in libimagecodec.quram.so
CVE-2025-55177	Meta	WhatsApp	Security Feature Bypass	Incomplete authorization of linked device synchronization messages
CVE-2025-10585	Google	Chrome	Memory corruption	Type confusion in V8
CVE-2025-38352	Google	Android	Memory corruption	Race condition in kernel
CVE-2025-48543	Google	Android	Memory corruption	Deserialization of untrusted data in ART

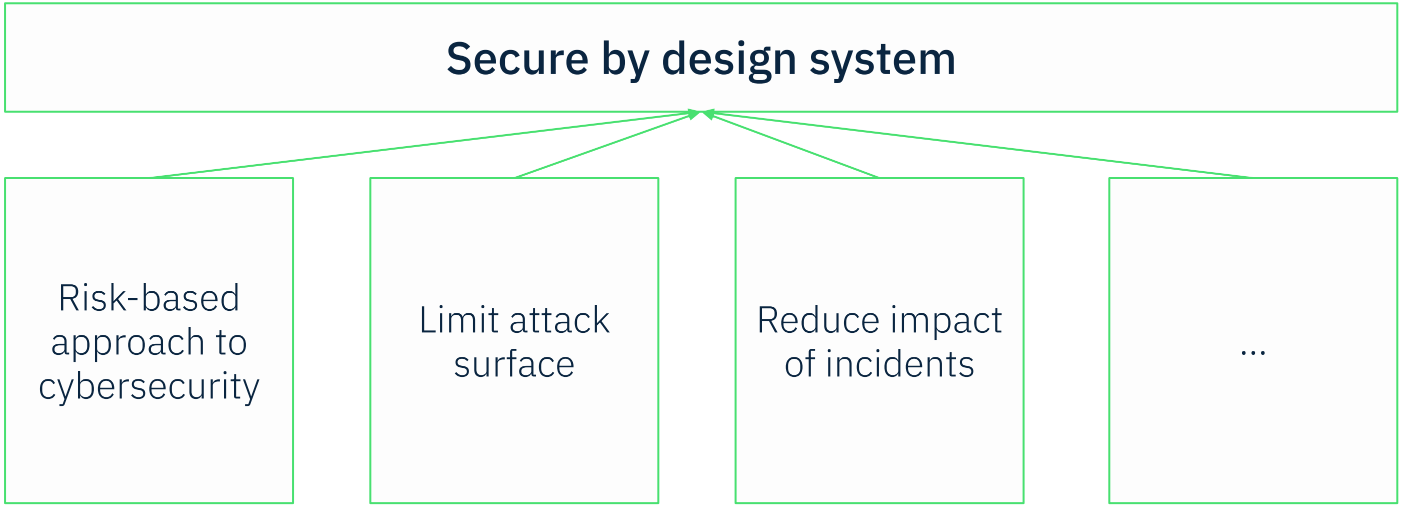
Google Project Zero finds ~70% as well



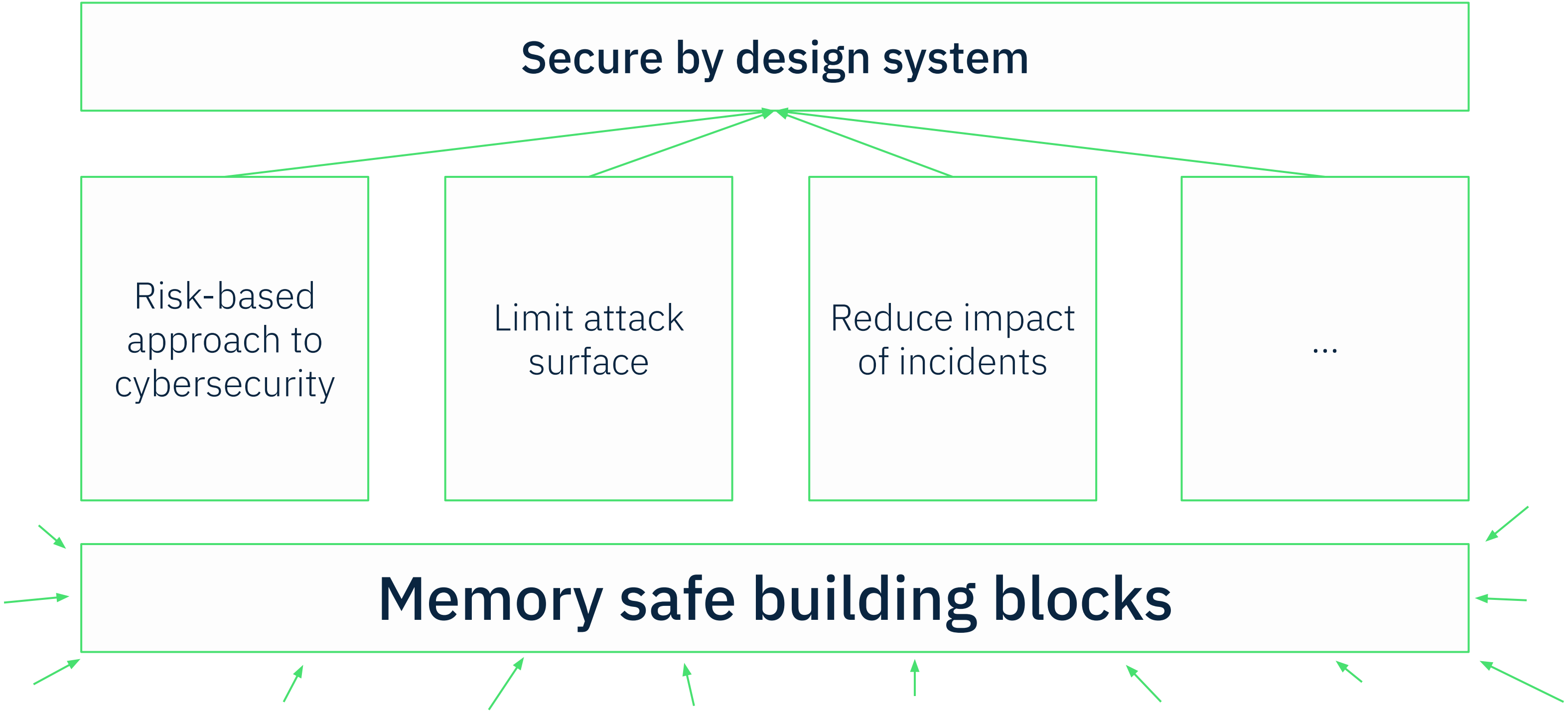
26 out of 36 0-days are memory safety **vulnerabilities!**

72%!

Using memory safe technology is foundational


Secure by design system
Risk-based approach to cybersecurity
Limit attack surface
Reduce impact of incidents
...

Using memory safe technology is foundational



Secure by design system

Risk-based approach to cybersecurity

Limit attack surface

Reduce impact of incidents

...

Memory safe building blocks

What is memory safe technology?

- Memory safety is about **programming languages**
- **Building blocks** of our digital systems
- **Built-in protection** against accidentally mishandling memory access
- **Memory safe**: Rust, Swift and Go, ...
- **Memory *unsafe***: like C and C++

Memory *un*safety in the wild

- **Memory unsafe code is everywhere!**
- Operating systems such as **Windows, macOS, iOS and Android**
- **Microsoft Office** is largely written in C++
- **VPNs**
- **Screen sharing solutions**
- Apps like **Zoom** and **Teams**

Memory safety vulnerabilities

- Remain despite developer training and tooling
- Hard to detect
- Often used in exploits
- Costly to fix

The impact is real, also in The Netherlands

Public Prosecution Service: compromised

- Severe disruption for many weeks
- Due to CVE-2025-6543 in Citrix Netscaler
- *Common Weakness Enumeration* CWE-119:

  Improper Restriction of Operations within the Bounds of a Memory Buffer

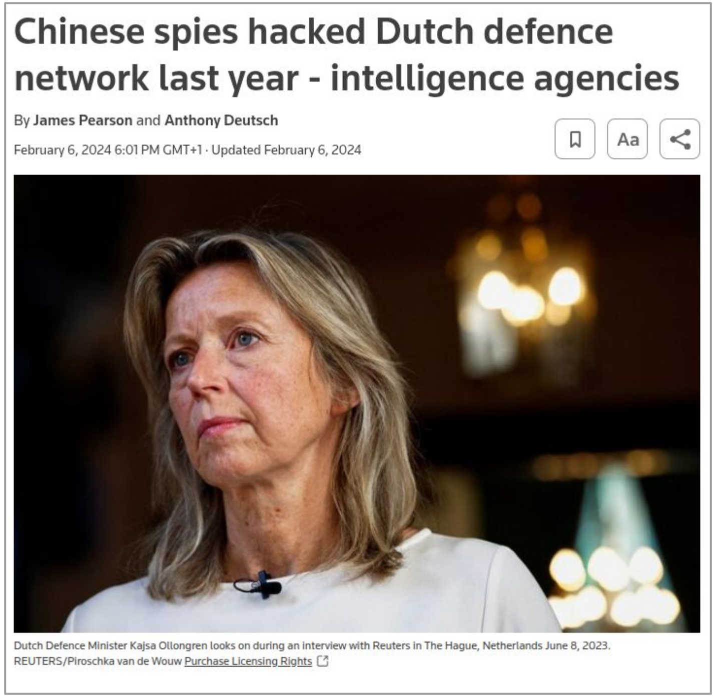
Openbaar Ministerie is inderdaad gehackt

Zware commissie gaat Citrix-lek bij OM onderzoeken

OPENBAAR MINISTERIE

Computable headlines (22 july 2025, 3 september 2025)

The Dutch Ministry of Defense hack



Chinese spies hacked Dutch defence network last year - intelligence agencies

By **James Pearson** and **Anthony Deutsch**

February 6, 2024 6:01 PM GMT+1 · Updated February 6, 2024

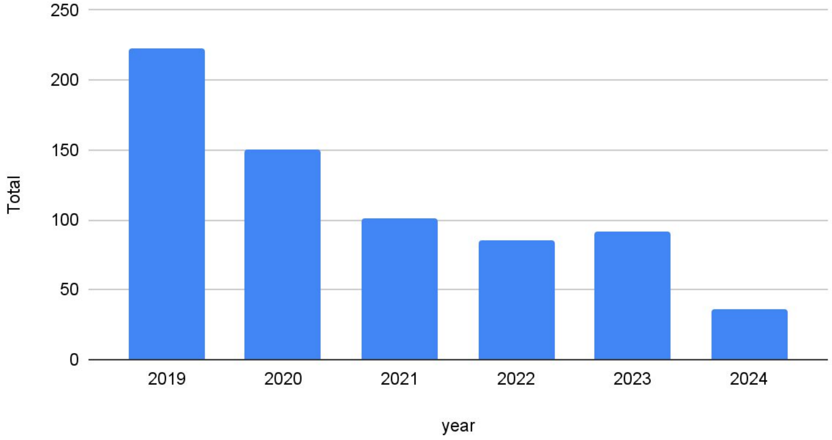
Dutch Defence Minister Kajsa Ollongren looks on during an interview with Reuters in The Hague, Netherlands June 8, 2023. REUTERS/Piroschka van de Wouw Purchase Licensing Rights

- COATHANGER malware
- CVE-2022-42475 in FortiGate
- **Heap-based buffer overflow**

Eliminating memory safety vulnerabilities in Android

- Problem is overwhelmingly with **new code**
- Using **memory safe technology** actually fixes the problem

Number of Memory Safety Vulns per Year



From 76% to 24% over 6 years

Despite the majority of code still being unsafe.

[Eliminating Memory Safety Vulnerabilities at the Source]

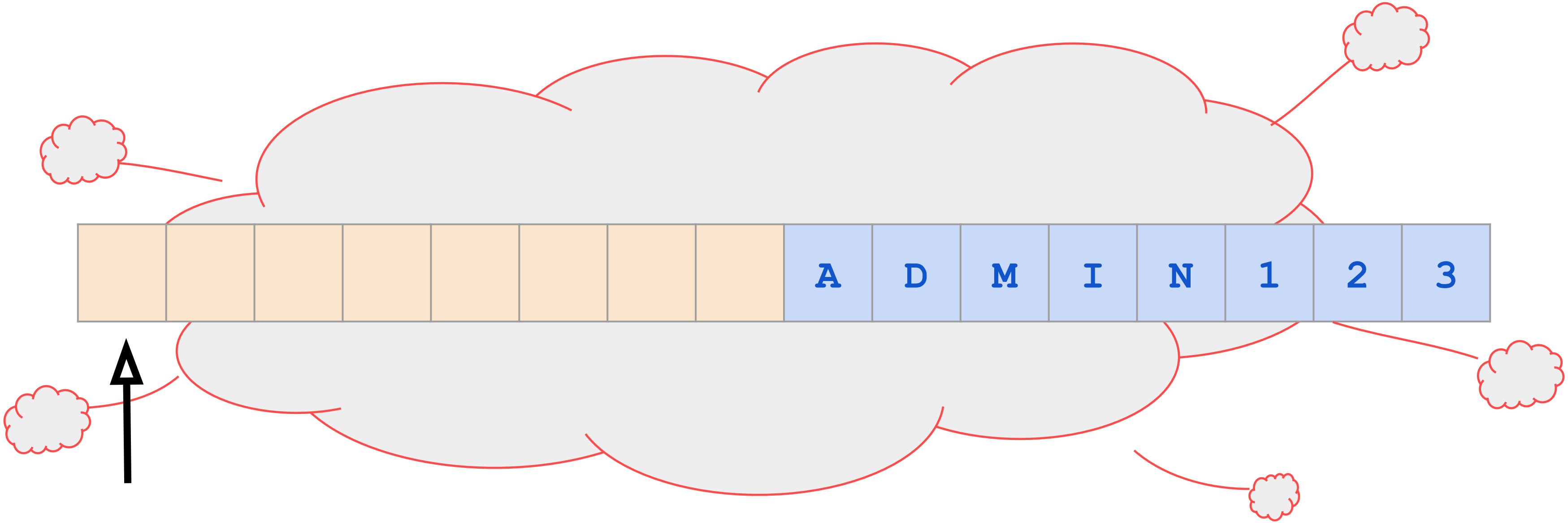
CWE-119, also known as: *buffer overflow, buffer overrun, …*

Leads to:

- Reading Unauthorized Data
- Crash
- Unexpected Results
- Execute Arbitrary Code

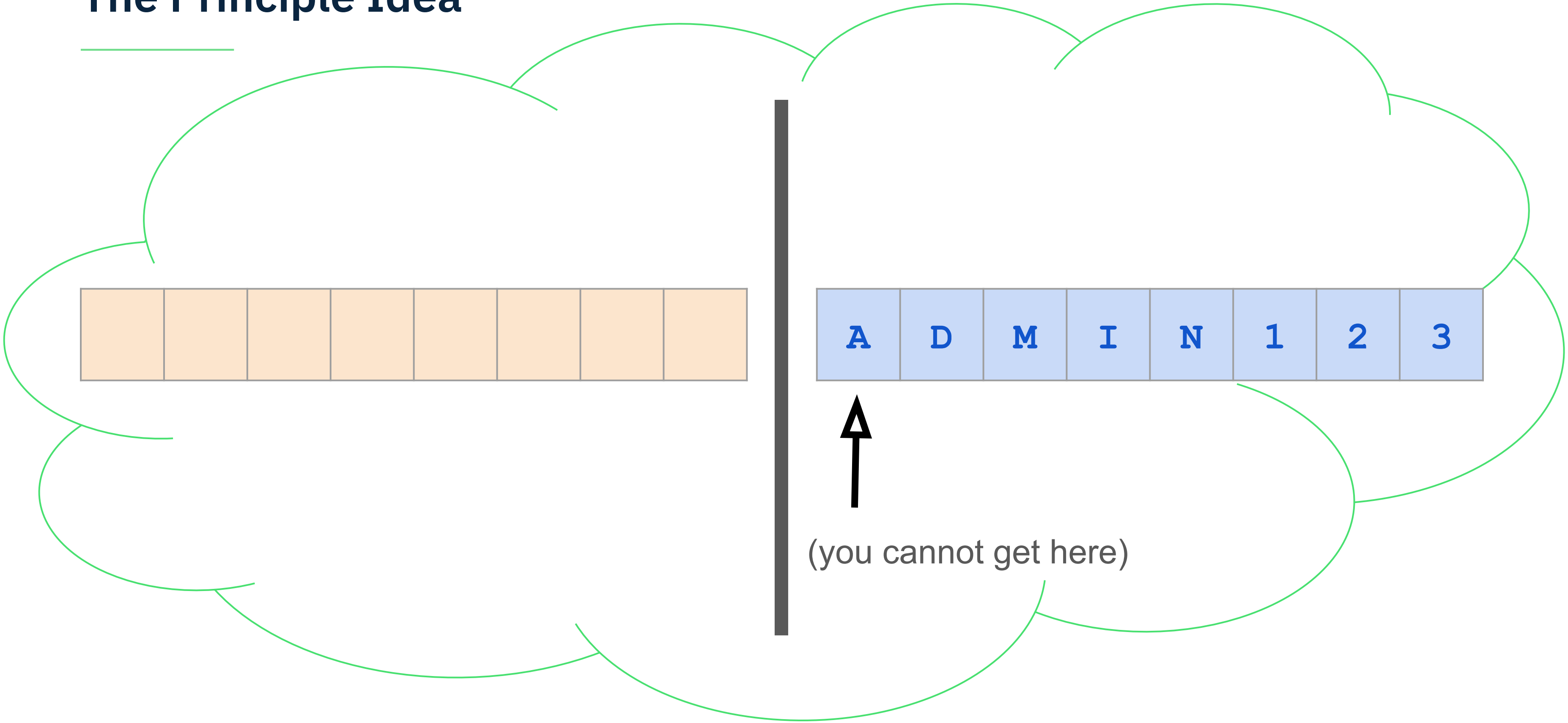
https://cwe.mitre.org/data/definitions/119.html

The Principle Idea



(you are here)

The Principle Idea



CWE-119, also known as: *buffer overflow, buffer overrun, ...*

Common consequences:

> *[...] If the attacker can overwrite a pointer's worth of memory (usually 32 or 64 bits), they can alter the intended control flow [...]*

Mitigation:

> ***Use a language that does not allow this weakness** to occur or provides constructs that make this weakness easier to avoid.*

https://cwe.mitre.org/data/definitions/119.html

Not just buffer overflows!

CWE-134: Externally-Controlled Format String

CWE-244: Heap Inspection

CWE-415: Double Free

CWE-416: Use After Free

CWE-690: NULL Dereference

CWE-824: Uninitialized Memory

https://cwe.mitre.org/data/definitions/1399.html

Attack the root cause

There are *mitigations* against this!

We can use *code analysis tools.*

But they can be stopped.

<u>At the source</u>.

Image credit: Andreas Weith (CC-BY-SA 4.0)

So I have to use their favorite programming language?

The Rust Programming Language

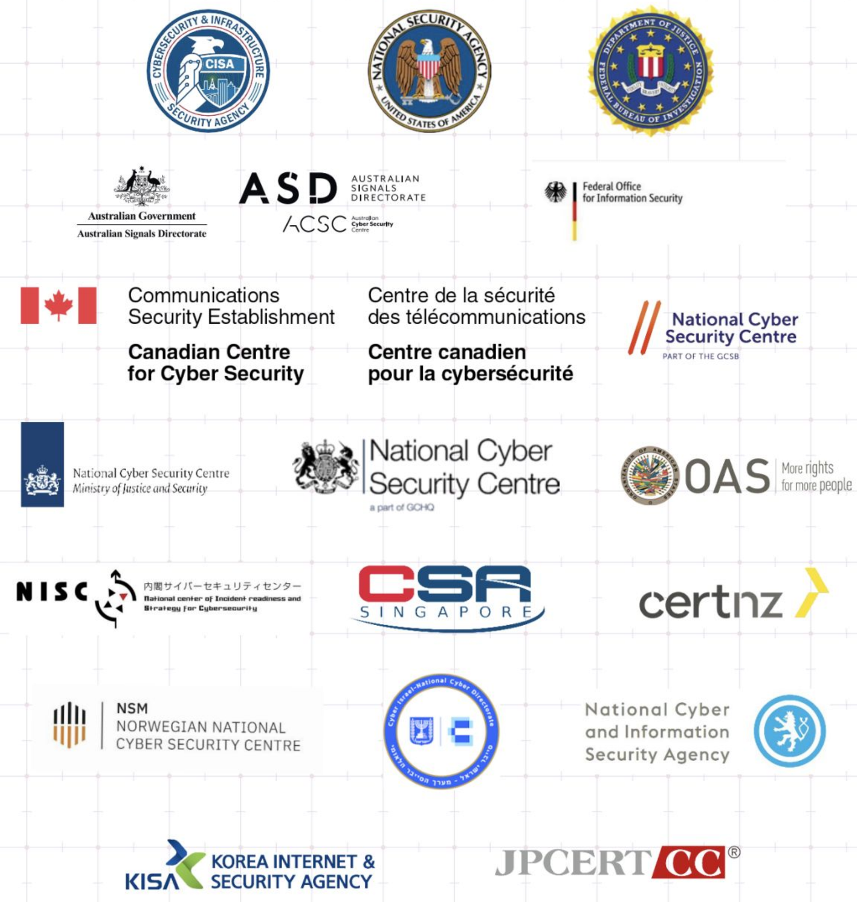
NO!

We’re not the only ones who care...!

- **CISA’s** campaign as of April 2023
- **7 publications about memory safety**
- Numerous co-authoring organizations
- Even the **White House** cares! (or used to...)

18 co-authoring organizations

- Our own NCSC!


CYBERSECURITY & INFRASTRUCTURE SECURITY AGENCY
CISA
NATIONAL SECURITY AGENCY
UNITED STATES OF AMERICA
DEPARTMENT OF JUSTICE
FEDERAL BUREAU OF INVESTIGATION
Australian Government
Australian Signals Directorate
ASD AUSTRALIAN SIGNALS DIRECTORATE
ACSC Australian Cyber Security Centre
Federal Office for Information Security
Communications Security Establishment
Canadian Centre for Cyber Security
Centre de la sécurité des télécommunications
Centre canadien pour la cybersécurité
National Cyber Security Centre
PART OF THE GCSB
National Cyber Security Centre
Ministry of Justice and Security
National Cyber Security Centre
a part of GCHQ
OAS
More rights for more people
NISC
内閣サイバーセキュリティセンター
National center of Incident readiness and Strategy for Cybersecurity
CSA
SINGAPORE
certnz
NSM
NORWEGIAN NATIONAL CYBER SECURITY CENTRE
Israel National Cyber Directorate
National Cyber and Information Security Agency
KISA
KOREA INTERNET & SECURITY AGENCY
JPCERT CC®

Publication highlights 1/4

TLP:CLEAR

SECURE BY DESIGN

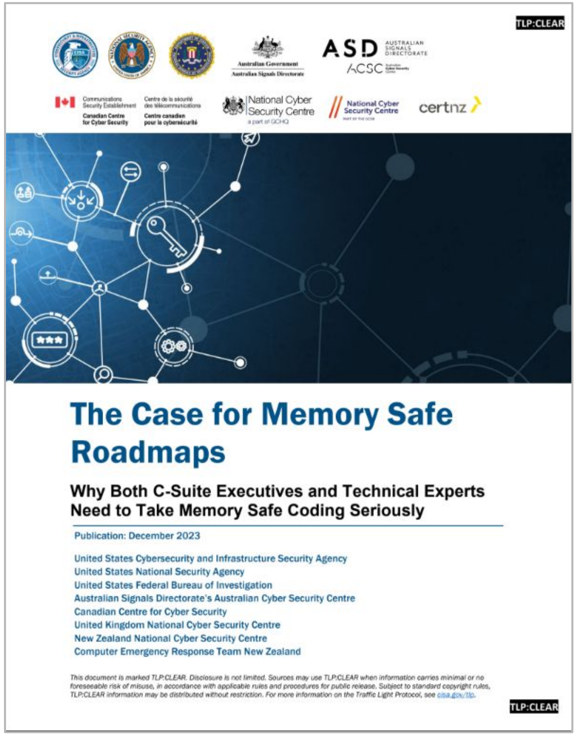
SHIFTING THE BALANCE OF CYBERSECURITY RISK:

PRINCIPLES AND APPROACHES FOR
SECURE BY DESIGN SOFTWARE

TLP:CLEAR

- The use of memory safe programming languages is the **#1 recommended tactic** (of 12 in total).

Publication highlights 2/4



TLP:CLEAR

The Case for Memory Safe Roadmaps

Why Both C-Suite Executives and Technical Experts Need to Take Memory Safe Coding Seriously

Publication: December 2023

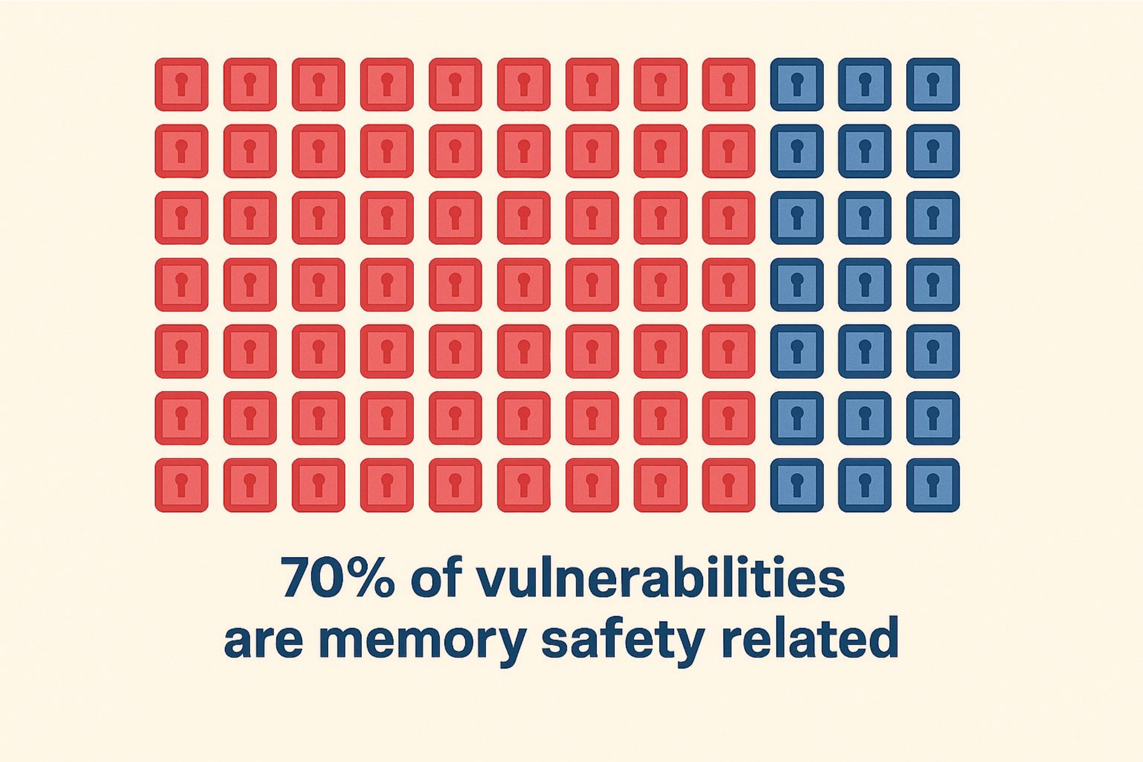
United States Cybersecurity and Infrastructure Security Agency
United States National Security Agency
United States Federal Bureau of Investigation
Australian Signals Directorate's Australian Cyber Security Centre
Canadian Centre for Cyber Security
United Kingdom National Cyber Security Centre
New Zealand National Cyber Security Centre
Computer Emergency Response Team New Zealand

This document is marked TLP:CLEAR. Disclosure is not limited. Sources may use TLP:CLEAR when information carries minimal or no foreseeable risk of misuse, in accordance with applicable rules and procedures for public release. Subject to standard copyright rules, TLP:CLEAR information may be distributed without restriction. For more information on the Traffic Light Protocol, see cisa.gov/tlp.

TLP:CLEAR

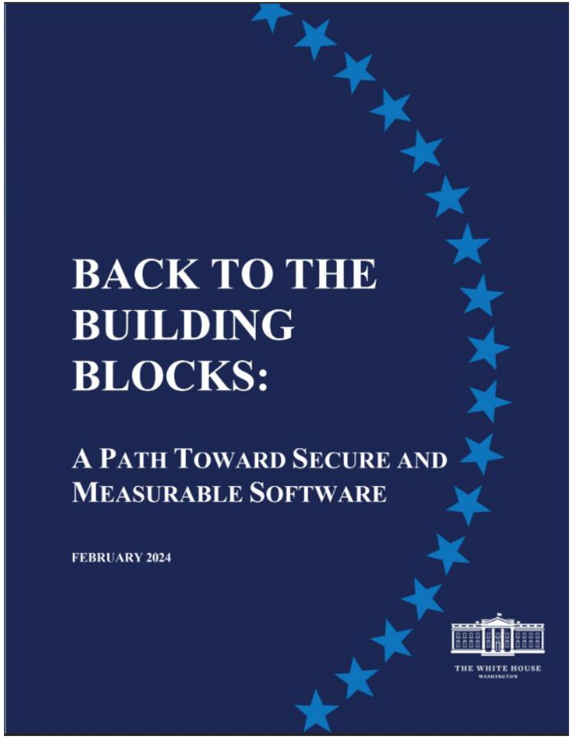
- “Eliminating this vulnerability class should be seen as a **business imperative**”

Remember, this class



70% of vulnerabilities
are memory safety related

Publication highlights 3/4



BACK TO THE BUILDING BLOCKS:

A PATH TOWARD SECURE AND MEASURABLE SOFTWARE

FEBRUARY 2024

THE WHITE HOUSE

WASHINGTON

- "[...] **memory safe hardware and formal methods** can be excellent complementary approaches"
- "**one of the most impactful actions** [...] is adopting memory safe programming languages."

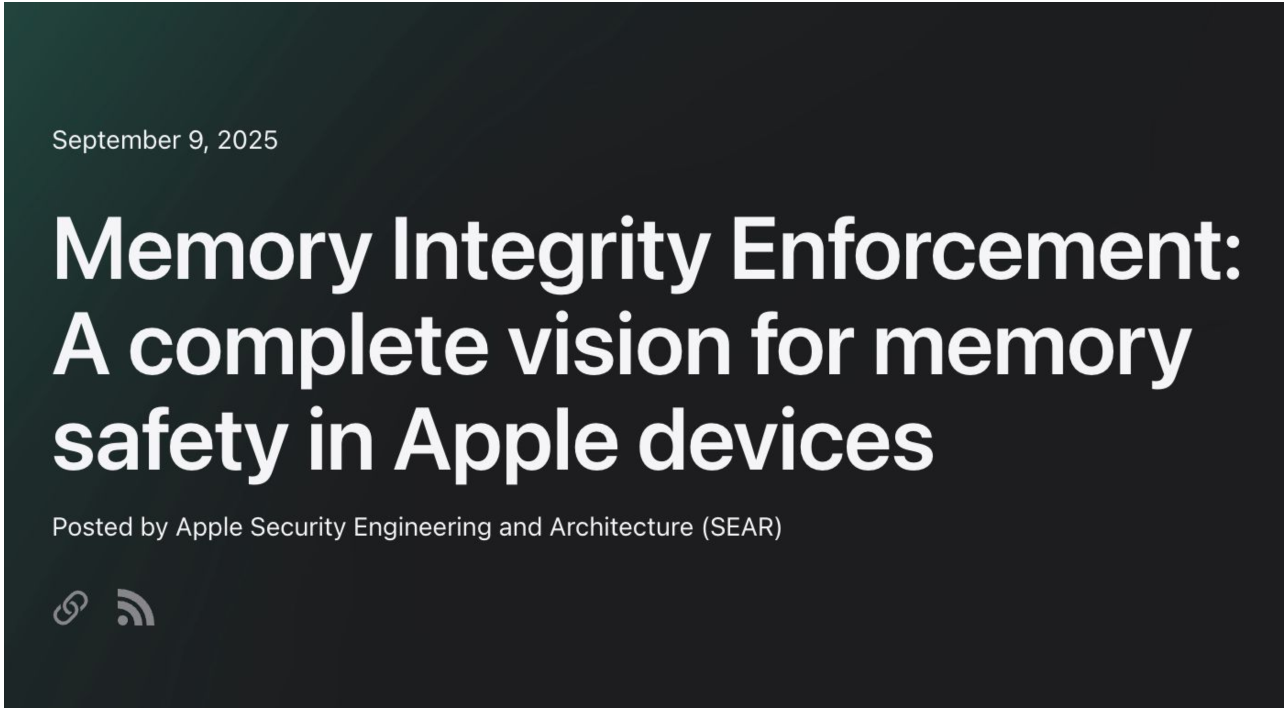
Publication highlights 4/4

Secure by Design Alert: Eliminating Buffer Overflow Vulnerabilities

CYBERSECURITY & INFRASTRUCTURE SECURITY AGENCY
CISA
DEPARTMENT OF JUSTICE
FEDERAL BUREAU OF INVESTIGATION
TLP:CLEAR
SECURE BY
DESIGN

- CISA and FBI: buffer overflow vulnerabilities are **unforgivable defects**.
- “[...] the use of memory-unsafe programming languages [...] poses **unacceptable risk** to our national and economic security.”

Apple is in the game too



September 9, 2025

Memory Integrity Enforcement: A complete vision for memory safety in Apple devices

Posted by Apple Security Engineering and Architecture (SEAR)

What about the EU?

European publication coming up!

Improving Europe's cybersecurity posture through memory safety

Hugo van de Pol, **Trifecta Tech Foundation**
Tara Tarakiyee, **Sovereign Tech Agency**

Statement on GitHub

Investments will need to be made

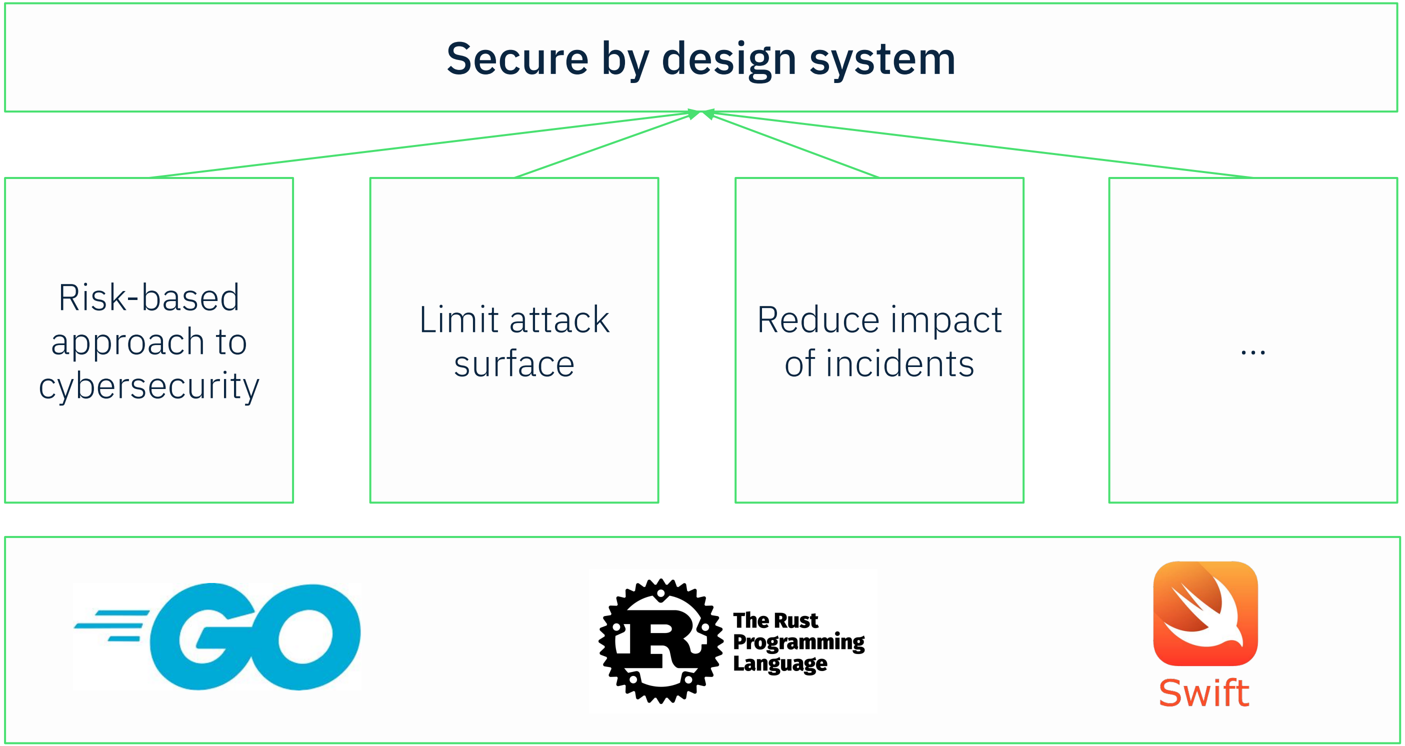
- Investigate their specific business case
- Build a roadmap for step-by-step adoption
- Focus on new systems / new code
- Upskill engineers

Investments will need to be made

- Investigate their specific business case
- Build a roadmap for step-by-step adoption
- Focus on new systems / new code
- Upskill engineers

An ounce of prevention is worth a pound of cure

There is a world to gain!



Secure by design system

Risk-based approach to cybersecurity

Limit attack surface

Reduce impact of incidents

…

What can you do?

- Help spread the word!
- Policy makers:
  - The CRA standardization process
  - Conditions you set in procurement
- Industry:
  - Investigate your business case for incremental adoption
  - Grab the current momentum
  - **Will you still be fixing all those vulnerabilities 10 years from now?**

Thanks

Getting in touch

Contact us or check out Tweede Golf on https://tweedegolf.nl or LinkedIn

Hugo van de Pol

Director
hugo@tweedegolf.nl
hugovandepol.bsky.social

Marc Schoolderman

Systems software engineer
marc@tweedegolf.nl

tweede golf

Further reading

1. Browse the CRA:
   http://www.european-cyber-resilience-act.com
2. Insight into the CRAs standardization process:
   https://fluchsfriction.medium.com/cyber-resilience-act-when-will-requirements-finally-get-more-specific-a990cc1dab24
3. NCSC Advisory Citrix NetScaler ADC and NetScaler Gateway:
   https://advisories.ncsc.nl/2025/ncsc-2025-0196.html
4. MIVD AIVD Advisory Coathanger:
   https://www.ncsc.nl/documenten/publicaties/2024/februari/6/mivd-aivd-advisory-coathanger-tlp-clear
5. Tracking sheet Google Project Zero
   https://googleprojectzero.github.io/0days-in-the-wild/rca.html

Further reading

6. Memory Safe Languages in Android 13
   https://security.googleblog.com/2022/12/memory-safe-languages-in-android-13.html
7. Eliminating Memory Safety Vulnerabilities at the Source
   https://security.googleblog.com/2024/09/eliminating-memory-safety-vulner...-Android.html

Bonus material

tweede golf

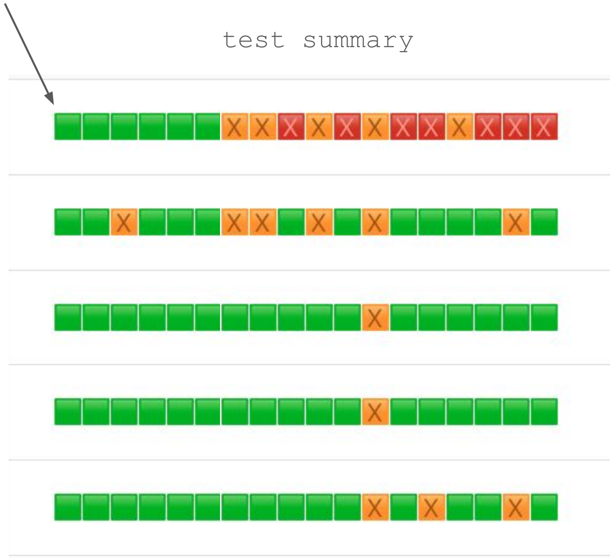
Our own experience

Background

- NCSC-2021-0982, a.k.a. **NAME:WRECK**
  https://advisories.ncsc.nl/2021/ncsc-2021-0982.html
- Affecting medical, industrial, aerospace, Iot devices.

Rewrite the defective code in Rust:

- Did engineers make mistakes? ***Yes***
- Did engineers cause vulnerabilities? ***No.***



memory unsafe language

test summary

🟩🟩🟩🟩🟩🟩🟧🟧🟥🟧🟥🟧🟥🟥🟧🟥🟥🟥

🟩🟩🟧🟩🟩🟩🟧🟧🟩🟧🟩🟧🟩🟩🟩🟩🟧🟩

🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩🟧🟩🟩🟩🟩🟩🟩

🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩🟧🟩🟩🟩🟩🟩🟩

🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩🟧🟩🟧🟩🟩🟧🟩

https://tweedegolf.nl/en/blog/152/does-using-rust-really-make-your-software-safer