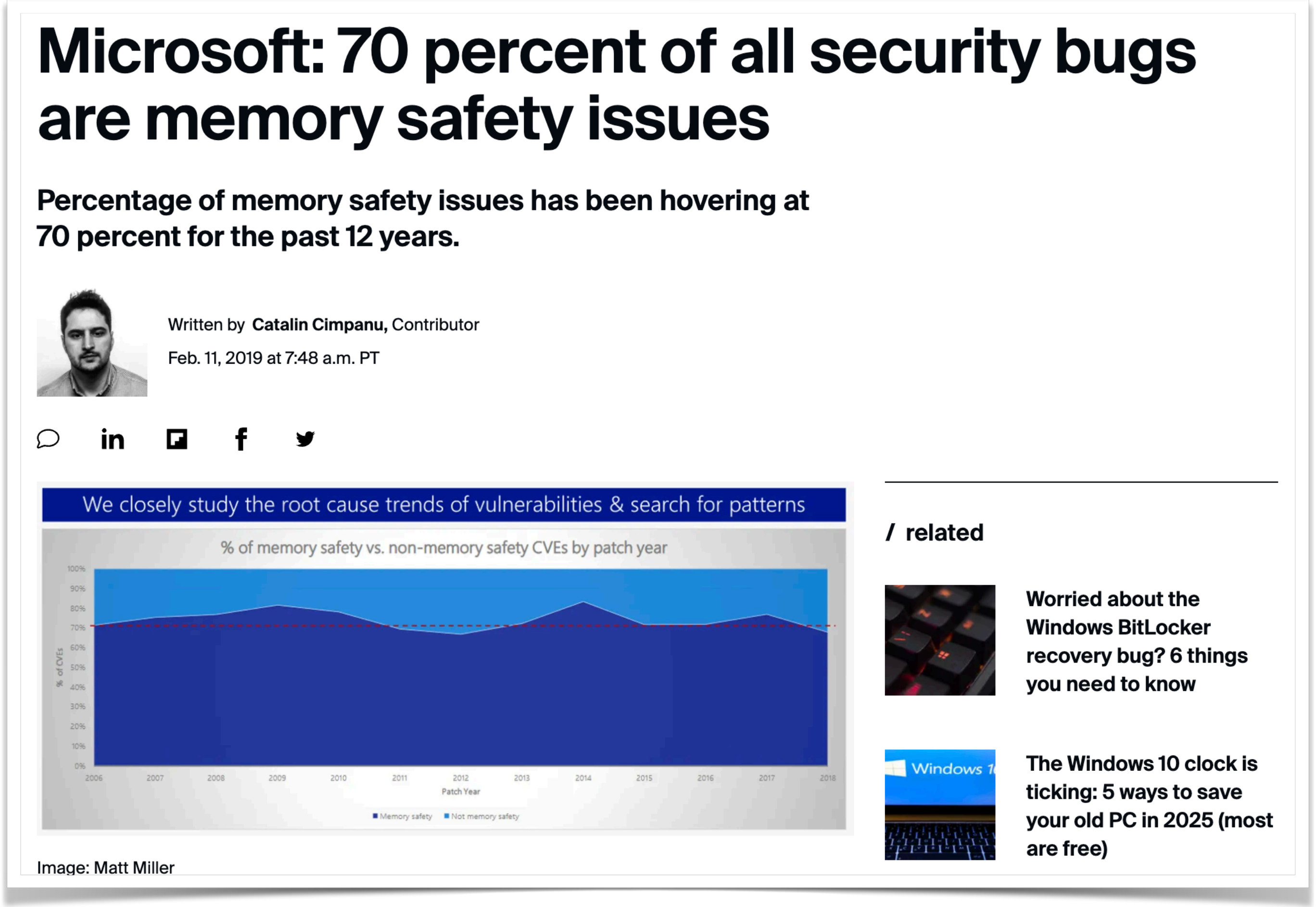
Memory safety
& Programming Languages

KC Sivaramakrishnan
kcsrk@cse.iitm.ac.in

Memory Safety

- A system is memory safe when it is devoid of *memory-related errors*
  - Buffer overflows
  - Use-after-free
  - Out-of-bounds access
  - Null pointer deference
  - …
- Memory unsafety leads to *undefined behaviours*
  - Opens doors to *security vulnerabilities*

¬Memory Safety ⟹ ¬Security



Microsoft: 70 percent of all security bugs are memory safety issues

Percentage of memory safety issues has been hovering at 70 percent for the past 12 years.

Written by **Catalin Cimpanu,** Contributor
Feb. 11, 2019 at 7:48 a.m. PT

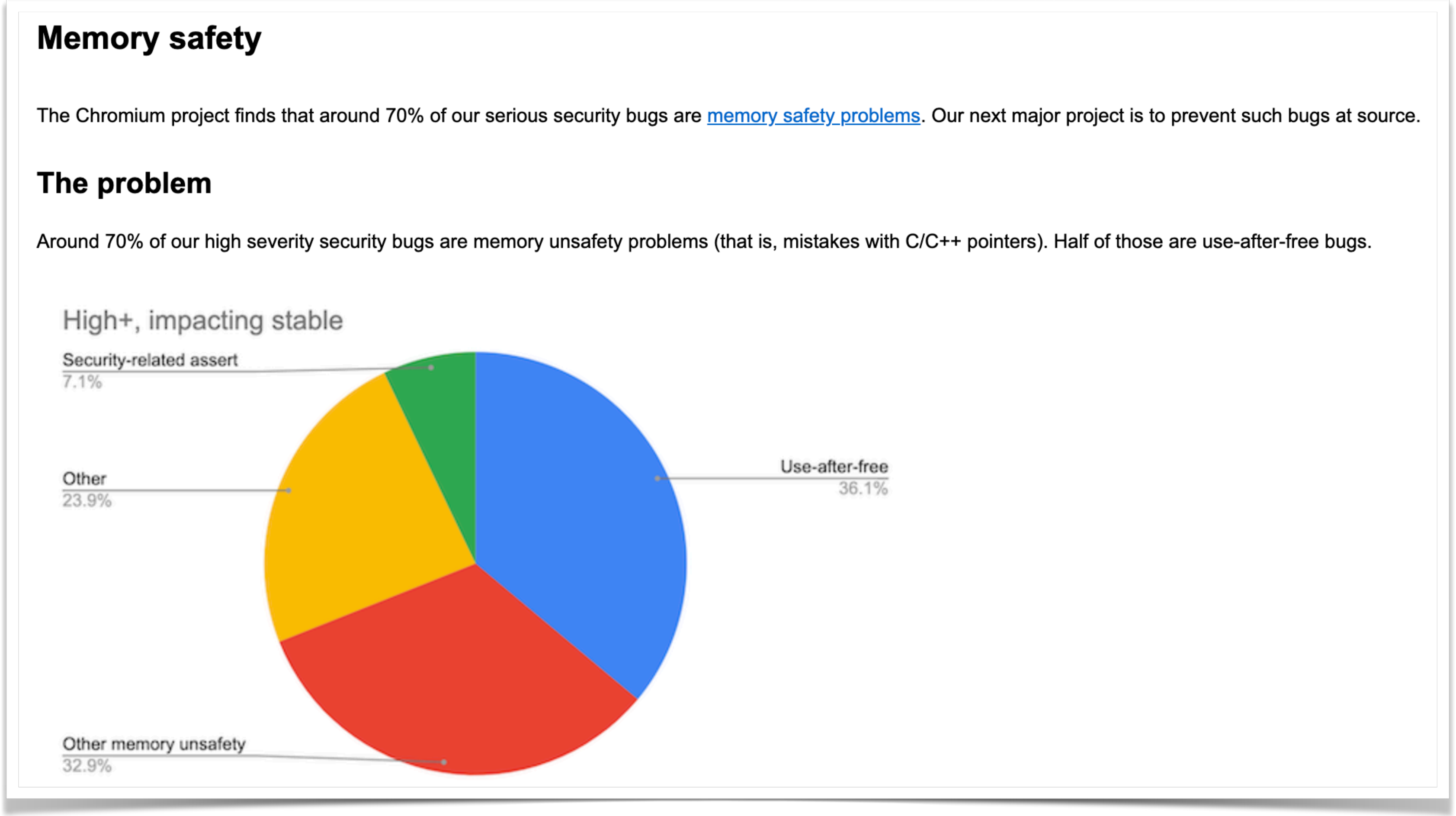
We closely study the root cause trends of vulnerabilities & search for patterns
% of memory safety vs. non-memory safety CVEs by patch year
Patch Year
Memory safety
Not memory safety

Image: Matt Miller

/ related

Worried about the Windows BitLocker recovery bug? 6 things you need to know

The Windows 10 clock is ticking: 5 ways to save your old PC in 2025 (most are free)



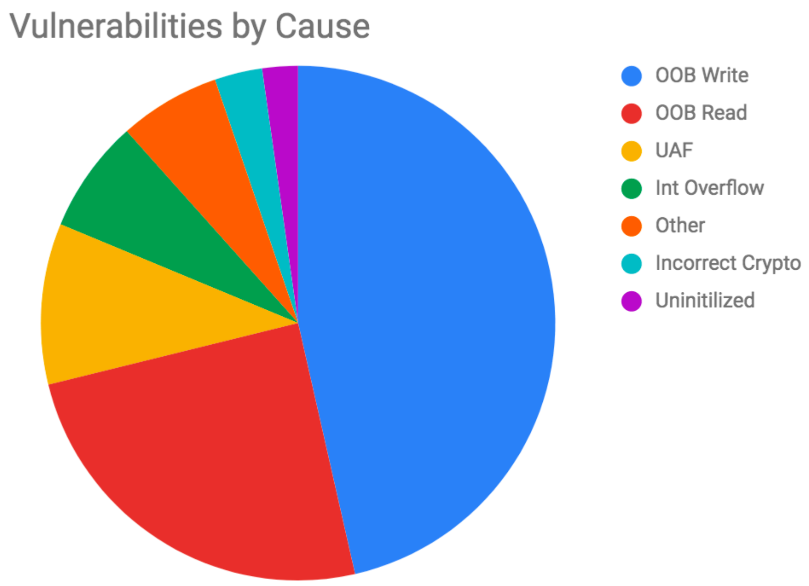
Memory safety

The Chromium project finds that around 70% of our serious security bugs are memory safety problems. Our next major project is to prevent such bugs at source.

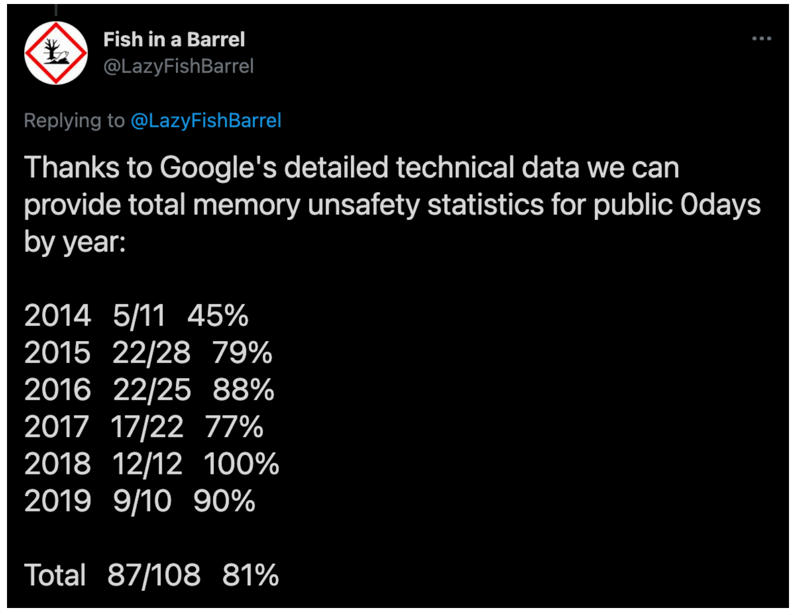
The problem

Around 70% of our high severity security bugs are memory unsafety problems (that is, mistakes with C/C++ pointers). Half of those are use-after-free bugs.

¬Memory Safety ⟹ ¬Security



90% of Android vulnerabilities are memory safety issues



Fish in a Barrel
@LazyFishBarrel

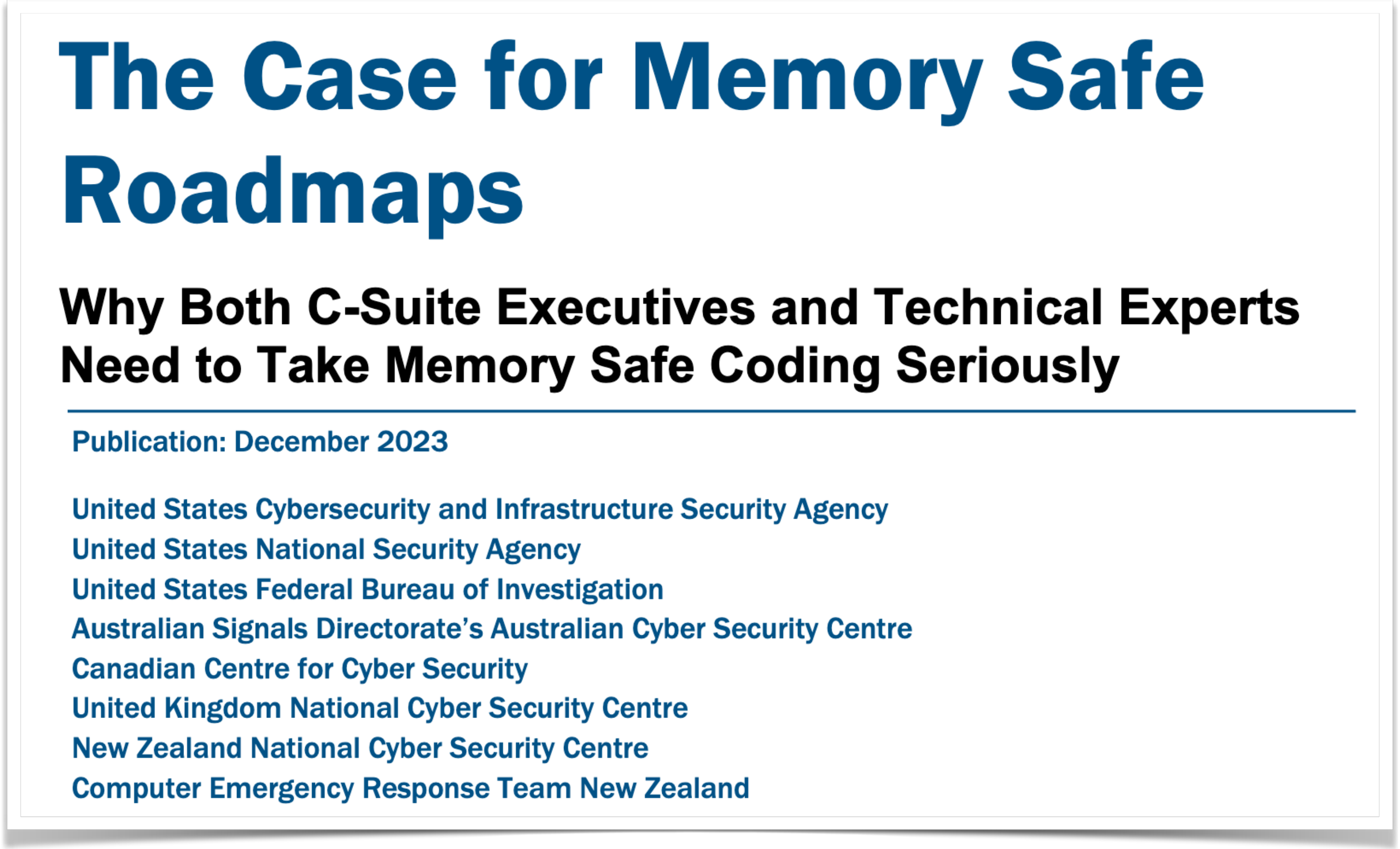
Replying to @LazyFishBarrel

Thanks to Google's detailed technical data we can provide total memory unsafety statistics for public 0days by year:

2014	5/11	45%
2015	22/28	79%
2016	22/25	88%
2017	17/22	77%
2018	12/12	100%
2019	9/10	90%
Total	87/108	81%

80% of the exploited vulnerabilities of known 0-days were memory safety issues

Memory Safety Recommendations

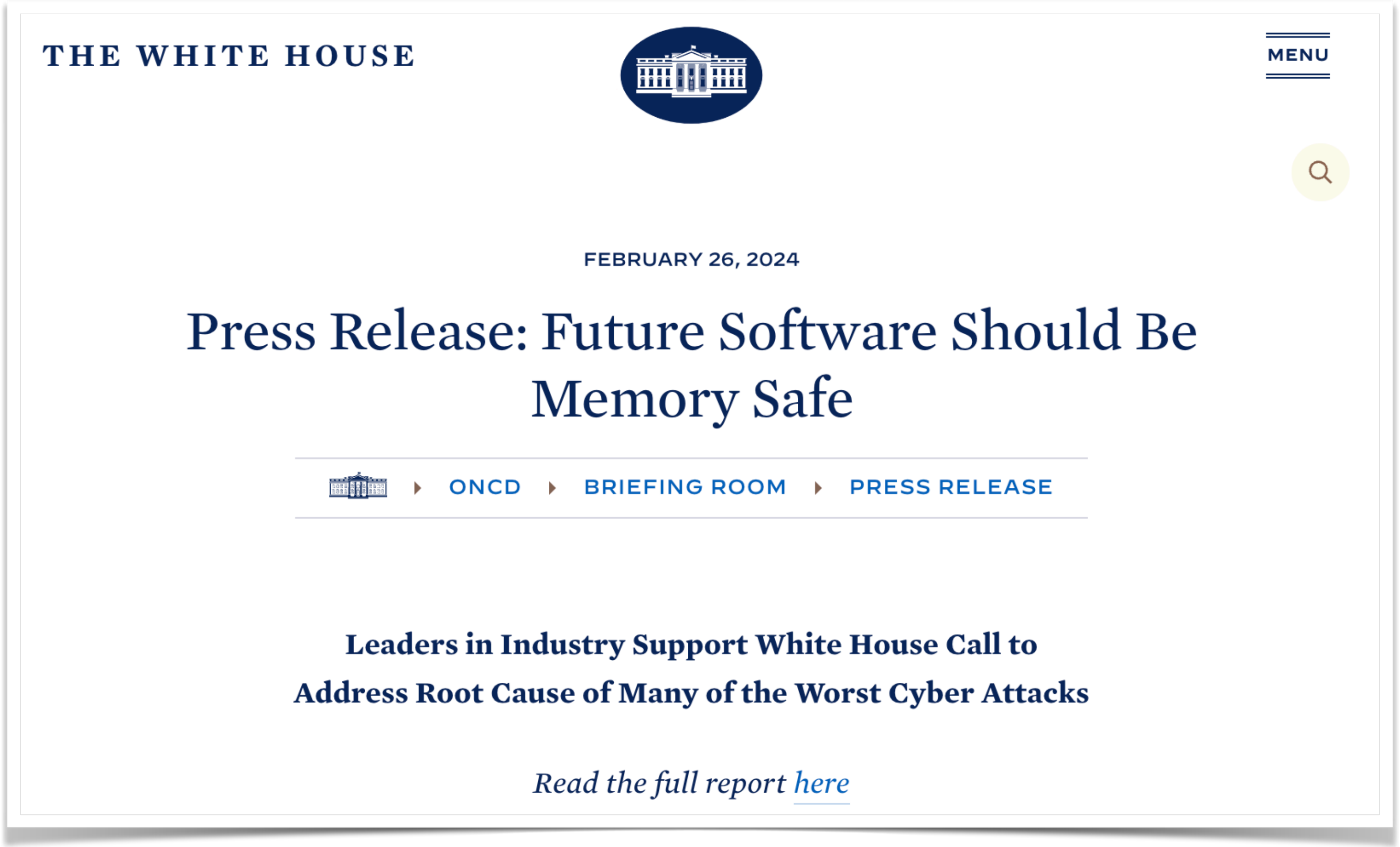


The Case for Memory Safe Roadmaps

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



FEBRUARY 26, 2024

Press Release: Future Software Should Be Memory Safe

Leaders in Industry Support White House Call to Address Root Cause of Many of the Worst Cyber Attacks

Read the full report here

1. Use memory-safe languages (primary)
2. Formally verify software (support)
3. Use secure hardware (support)

Memory Safety and Programming Languages

- Unsafe languages
  - C, C++, Assembly, Objective-C
- Safe languages
  - With the help of a garbage collector (GC) — JavaScript, Python, Java, Go, OCaml, …
  - Without a GC — Rust
- Unsafe parts of safe languages
  - Unsafe Rust, unsafe package in Go, Obj in OCaml

Safe-by-construction programming language

OCaml

industrial-strength, pragmatic, functional programming language

Industry

FACEBOOK Tarides Bloomberg

docker ahrefs SimCorp

Microsoft Tezos Jane Street

Projects

hack flow

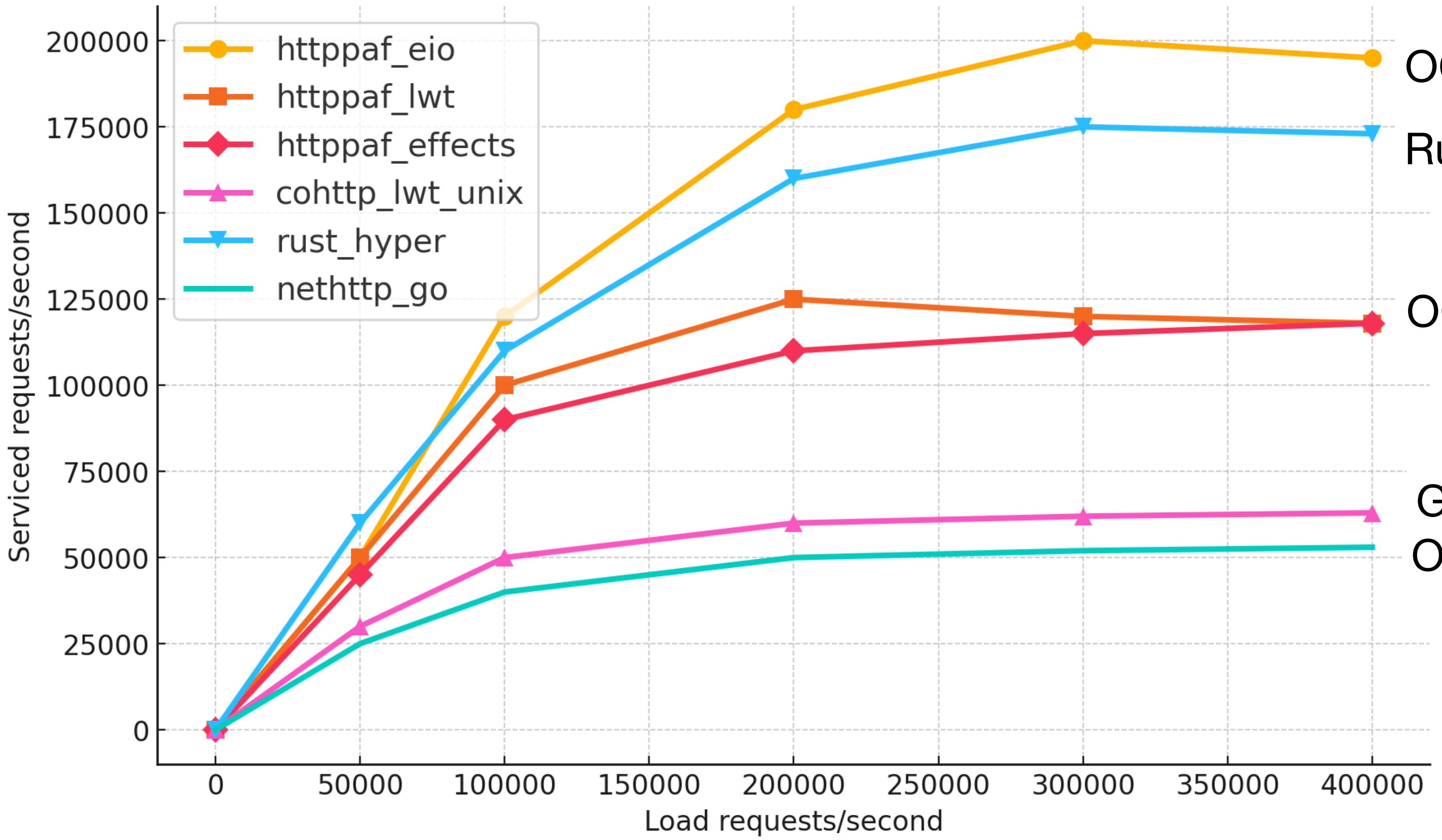
COMPCERT

20% of Wall Street trade goes through OCaml

Functional core with imperative and object-oriented features

Native (x86, Arm, Power, RISC-V), JavaScript, WebAssembly

OCaml Performance — Web Server



OCaml eio

Rust Hyper

OCaml (Http/af + Lwt)

Go NetHttp

OCaml (cohttp + Lwt)

https://github.com/ocaml-multicore/eio

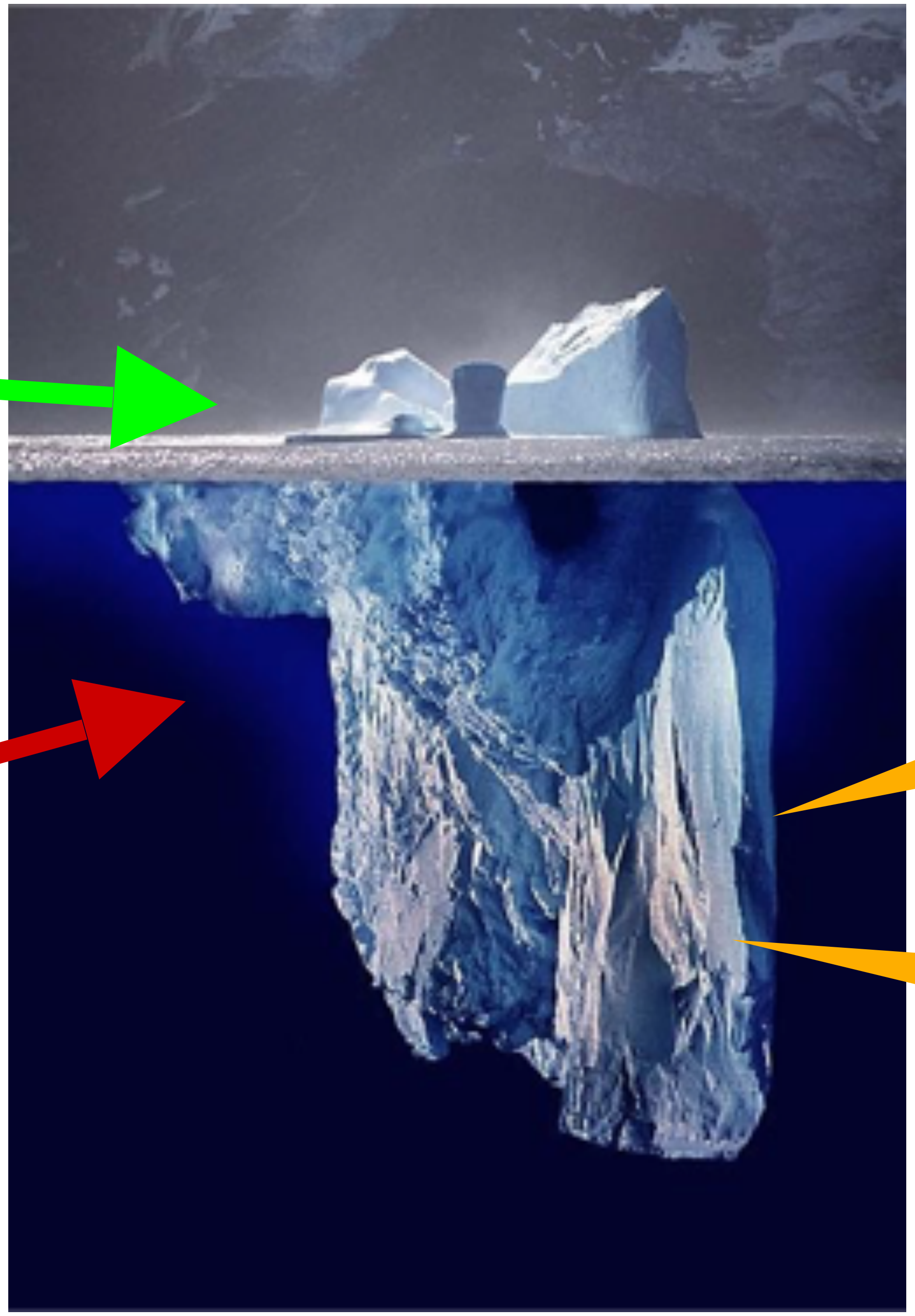
How far can we push this?

How about an entire OS?

Why? Monolithic OS Icebergs

Code you want to run

Code your operating system insists you need!

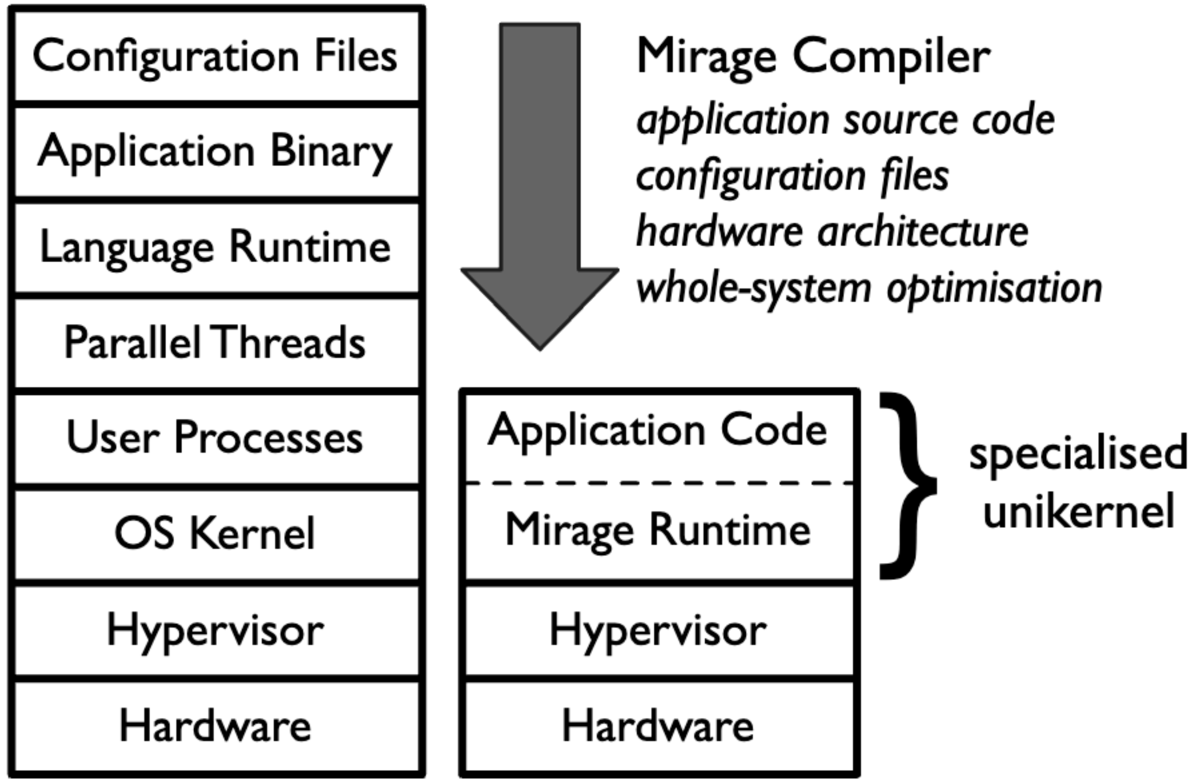


Huge TCB ⟹ Security concern

Written in memory-unsafe languages

MirageOS Unikernels

- MirageOS is a **library OS** to build specialised **Unikernels** containing only what is needed by the application
    - Cut the complexity by designing the layers as independent memory-safe libraries.
- Rely on the OCaml for memory safety, modular static analysis, dead-code elimination, etc.
- Used in
    - Docker for Mac and Windows
    - NetHSM — hardware security modules
    - SpaceOS
- See mirage.io



Configuration Files

Application Binary

Language Runtime

Parallel Threads

User Processes

OS Kernel

Hypervisor

Hardware

Mirage Compiler

application source code

configuration files

hardware architecture

whole-system optimisation

Application Code

Mirage Runtime

Hypervisor

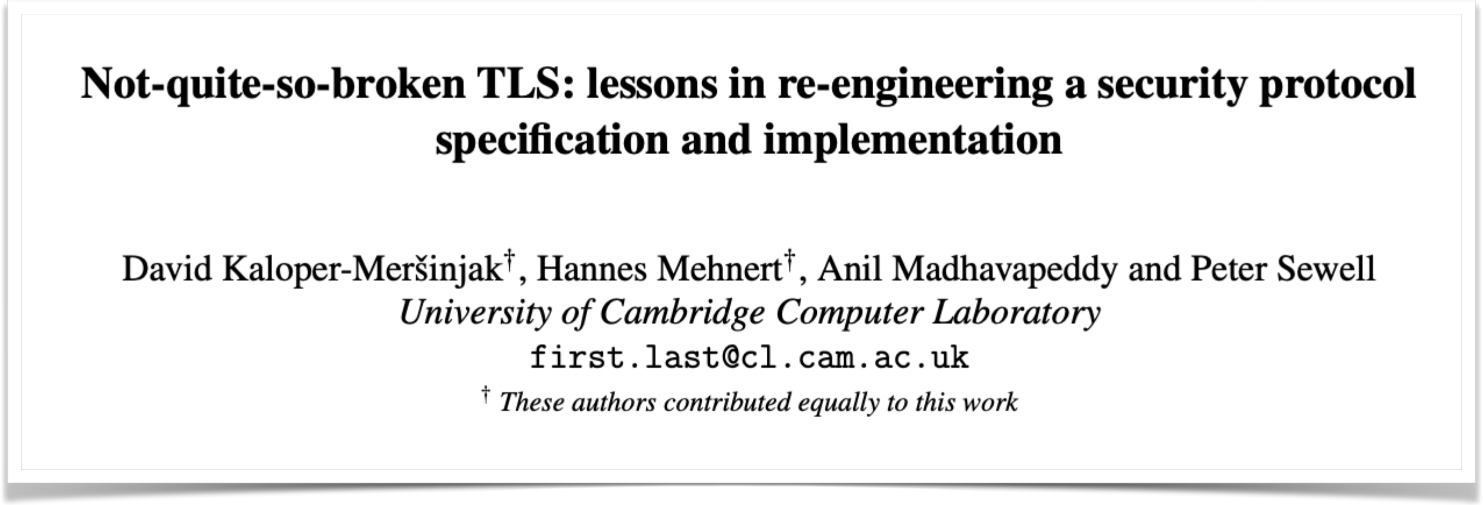
Hardware

specialised unikernel

Available Libraries

Network:
Ethernet, IP, UDP, TCP, HTTP 1.0/1.1/2.0, ALPN, DNS, ARP, DHCP, SMTP, IRC, cap-n-proto, emails
Storage:
block device, Ramdisk, Qcow, B-trees, VHD, Zlib, Gzip, Lzo, Git, Tar, FAT32
Data-structures:
LRU, Rabin's fingerprint, bloom filters, adaptative radix trees, discrete interval encoding trees
Security:
x.509, ASN1, TLS, SSH
Crypto:
hashes, checksums
Ciphers (AES, 3DES, RC4, ChaCha20/Poly1305)
AEAD primitives (AES-GCM, AES-CCM)
Public keys (RSA, DSA, DH)
Fortuna

- *Reimplemented in OCaml*
- TLS: "rigorous engineering"
  - same pure code to generate test oracles, verify oracle against real-world TLS traces and the real implementation
  - Use Fiat (Coq extraction) for crypto primitives.

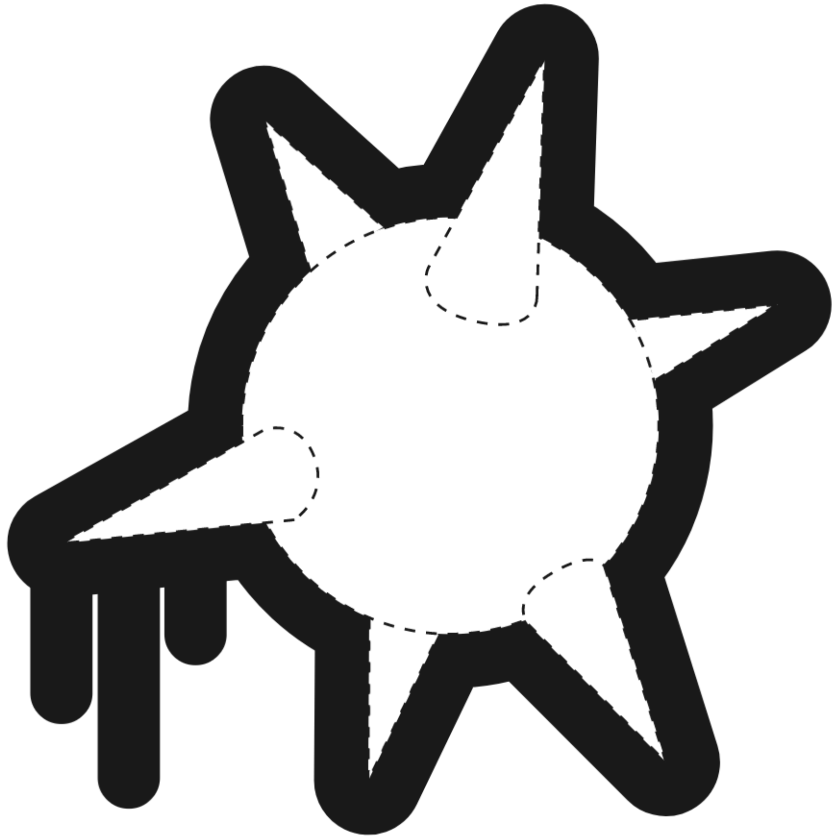


Not-quite-so-broken TLS: lessons in re-engineering a security protocol specification and implementation

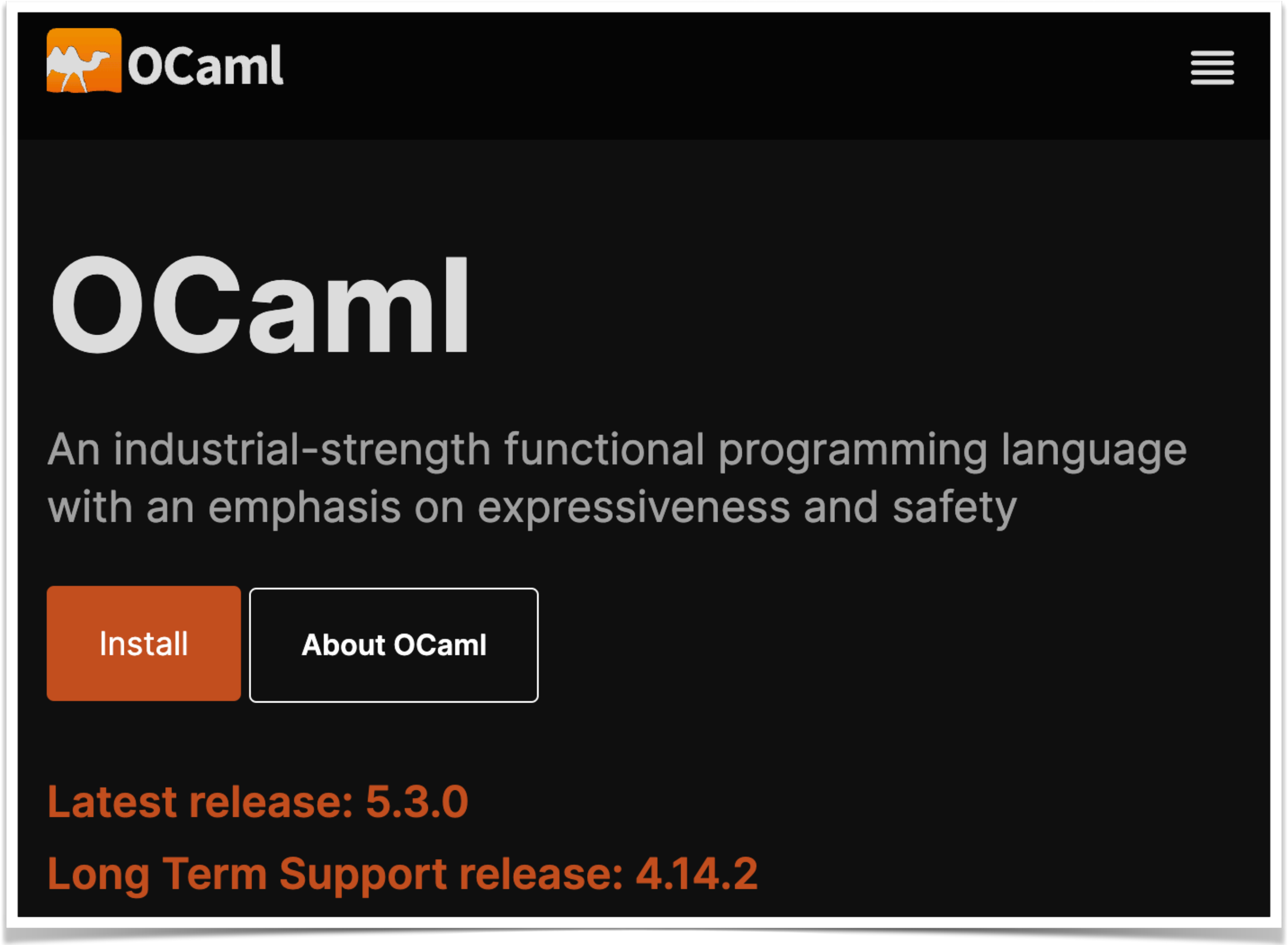
David Kaloper-Meršinjak[†], Hannes Mehnert[†], Anil Madhavapeddy and Peter Sewell
University of Cambridge Computer Laboratory
first.last@cl.cam.ac.uk
[†] *These authors contributed equally to this work*

Bitcoin Piñata

- https://hannes.robur.coop/Posts/Pinata
- 1.1 MB Unikernel, which ran from 2015 to 2018
- Hold the key to 10 bitcoins (peak worth $165k)
    - Now worth ~$1M
- A successful authenticated TLS session reveals the private Bitcoin key
- 500,000 accesses to the Piñata website, more than 150,000 attempts at connecting to the Piñata bounty
- The bitcoins were safe!



ocaml.org



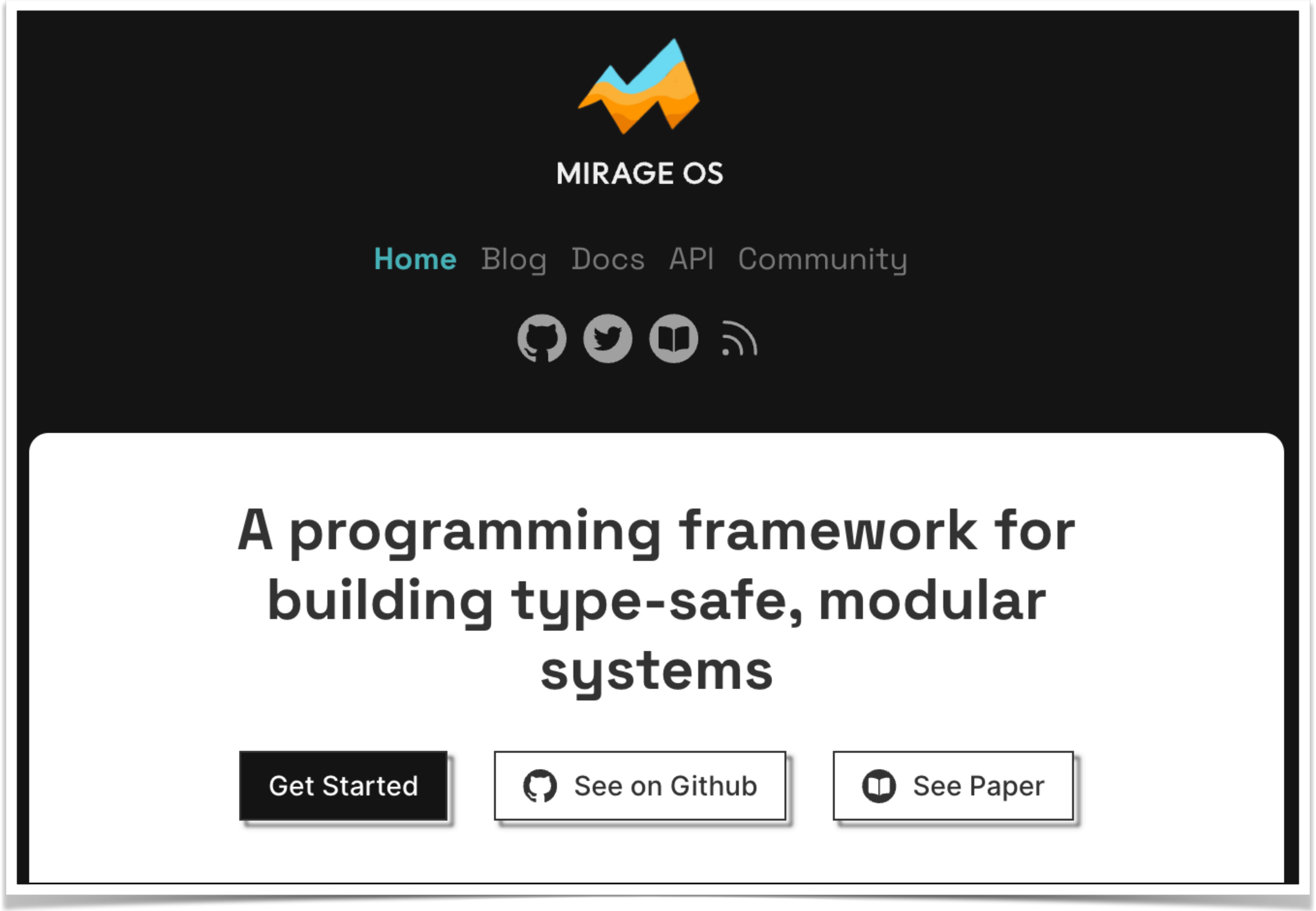
OCaml

An industrial-strength functional programming language with an emphasis on expressiveness and safety

Latest release: 5.3.0

Long Term Support release: 4.14.2

mirage.io



A programming framework for building type-safe, modular systems