CHERI technology overview

A QUICK INTRODUCTION

Data breaches are very costly

>$500M

Cost of addressing Heartbleed buffer overflow vulnerability

6X

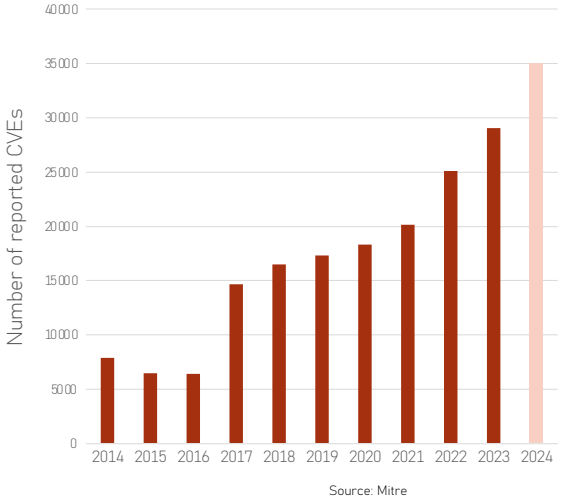
Increase in firmware attacks reported by NIST between 2017 and 2024

~$10T

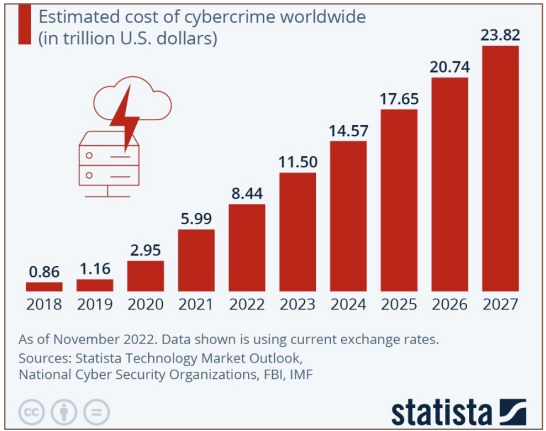
Worldwide cost estimate of cyberattacks per year

(and growing fast)

Vulnerabilities are causing increasing pain



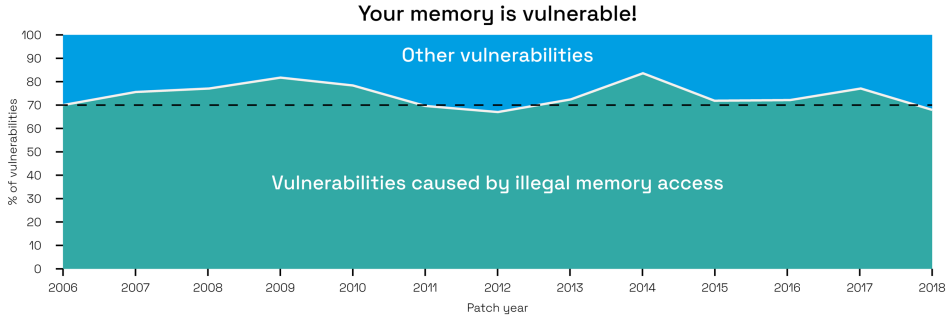
Source: Mitre



Memory safety is necessary

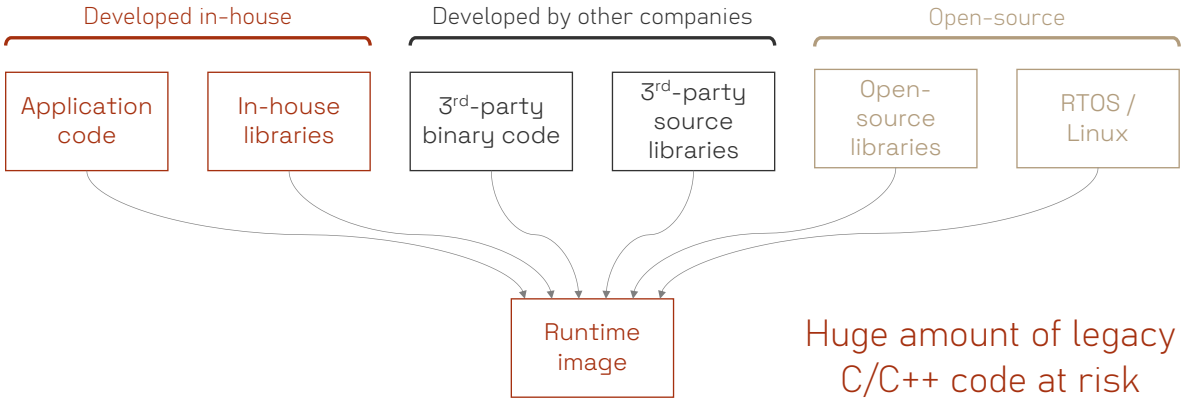
- Memory abuse (e.g. buffer overflows) is the main attack vector
- Constant ratio of over the past 20 years
  - … even with all the work done on software to avoid this!

CHERI solves this!



Source: "Trends, challenges and shifts in vulnerability mitigation", Matt Miller (MSRC), BlueHat IL 2019.

Impossible to re-write software to fix the problem



Huge amount of legacy C/C++ code at risk

Possible solutions for memory safety

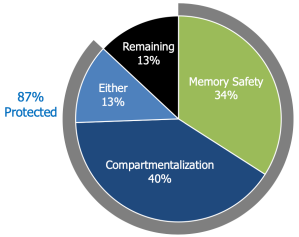
- ✗ Use “memory safe” languages like Rust or .Net
  - Requires rewriting trillions of lines of C/C++ code
  - Possible for new code, but no compartmentalisation
- ✗ Use “coarse-grained” techniques like stack “canaries” to detect issues
  - Helpful, but they statistically leave too many holes
  - Hacking techniques already developed
- ✓ Use “fine-grained” techniques like CHERI
  - Best option, but needs new hardware

Memory safety becomes a key topic

DARPA

Memory safety and compartmentalization would stop most campaigns

Mitigation analysis of Linux kernel high severity CVEs



87% Protected

- Either: if memory safety or compartmentalization were present, the bug would not be exploitable
- Remaining: memory safety or compartmentalization would not mitigate the CVE

CVE: Common Vulnerabilities and Exposures

McKee, Derrick, et al. "Preventing kernel hacks with HAKC." Proceedings 2022 Network and Distributed System Security Symposium. NDSS. Vol. 22. 2022.

5

Highlights CHERI as a solution

FEBRUARY 26, 2024

Press Release: Future Software Should Be Memory Safe

ONCD › BRIEFING ROOM › PRESS RELEASE

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

Introducing CHERI technology

C apability
H ardware
E nhanced
R ISC
I nstructions

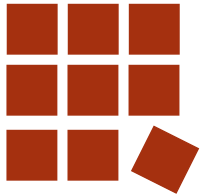
CHERI

Fine-grained memory protection

Revisits fundamental design choices in hardware and software to dramatically improve system security

Extend conventional hardware ISAs

- Memory protection
- Scalable compartmentalization



What makes CHERI different?

	Previous solutions	Problem	CHERI
Security control	Statistical • "likely" to detect a problem	Targeted attacks bypass protection Some problems detected too late	Systematic • 100% coverage
Enforcement	Some complex, disparate hardware features, no coherency across architectures Rely on "trusted" software and/or explicit checks	Additional complexity Software can be hacked (and has been…)	Hardware-enforced • Simple, holistic protection • No way to bypass by software (unforgeable tags)
Software impact	High impact on software • A lot of additional code needed to protect and isolate • Need best security experts to review all software stack • Often need to rewrite code	Difficult to catch all issues Reduce performance and increase code size Experts are scarce	Extremely low software impact • Need recompilation • Adapt some very low-level code
Type of solution	Reactive • Fix problem if vulnerability discovered Proactive solution possible (but uneconomical)	Huge exploitable attack surface, susceptible to 0-day attacks Most systems are not upgraded immediately / regularly	Preventive • Protects against existing and to-be-designed attacks on memory

Main memory issues with C/C++

- Possible to access memory out of expected buffers
  - Access confidential data
  - Modify / delete critical data or code
  - Inject malware code
  - Spy on communications
  - Erase traces of attack

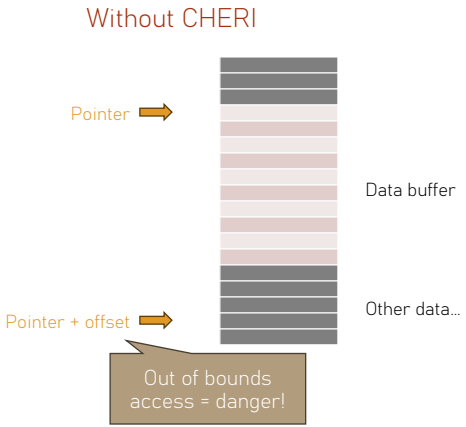
Need memory safety

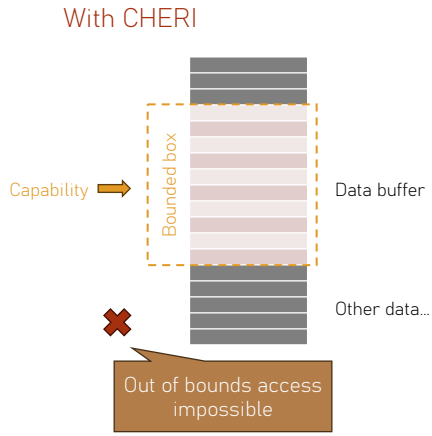
- Functions cannot protect their data from each other
  - Only works when the software can be trusted
  - Enable privilege escalation

Need compartmentalization

Spatial memory safety

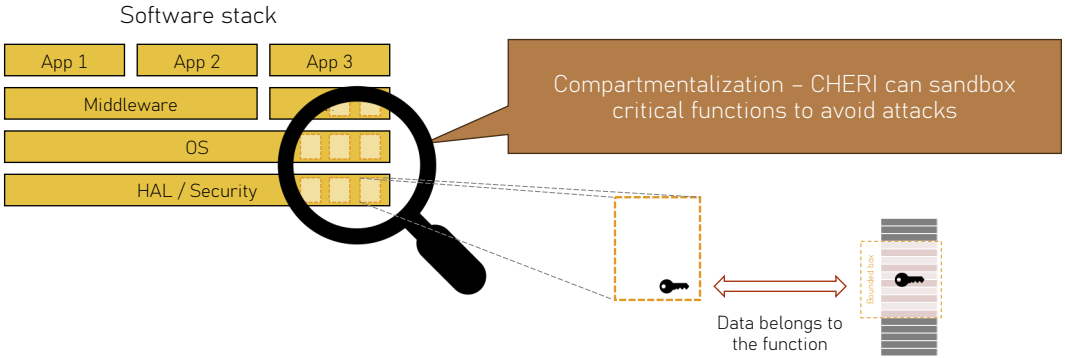
- Replacing pointers by capabilities – with hardware control





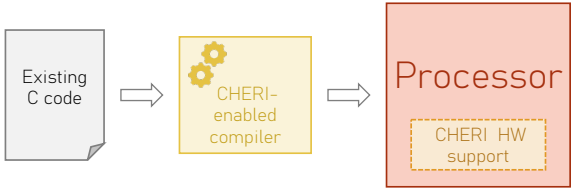
Compartmentalization

- Capabilities belong to an identified function / execution context



CHERI relies on hardware protection

- CHERI requires adapted processor
  - Can be applied to many types of core
- Hardware-enforced security
  - Impossible to bypass by software
  - Formally proven
- Reuse existing code
  - Just recompile application
  - Choose which part to protect
- Benefit from CHERI
  - Rejects dangerous code
  - Create CHERI compartments for critical code



CHERI has a positive impact

- Limited costs
  - Area impact
    - Processor only 4% larger*
    - Similar power consumption
    - Similar performance**
  - Memory impact
    - Small area for tag storage
    - Double size for pointers (mostly impacts stack)
    - No change on data storage requirements
  - Software development
    - Need adapted tools (open-source available)
    - Less than 0.5% application code*** to adapt
    - Need recompilation
    - OS / low-level drivers need work (but done once)

- Huge gains
  - Memory safety!
    - Save in patching costs
    - Compiler detects mistakes in existing code
  - Performance gains****
    - Remove or simplify software-based mechanisms (TEE, compartmentalization, security modes, …)
    - Eliminate context switching in hypervisor
    - Reduce code, improve execution speed
  - Fast, low-risk integration of unsafe code
  - Save security experts' time
    - Not wasting it on bug hunting…

CHERI Alliance will help by stimulating the ecosystem to adapt OS / software / tools

* Real data from Codasip – Comparing the same commercial processor with/without CHERI
** Slight degradation for chips with low-bandwidth internal bus (i.e. less than 128bit)
*** Real data from application porting
**** Partial OS / security mechanisms adaptation (done once, usually by the ecosystem)

CHERI has already got strong supporters

CHERI projects

- A number of prototypes / proof of concept have been released
  - Proof of concept

    
    arm Morello Program
  - Open-source / prototype

  - Commercial

- Some OS have been ported to CHERI (Free RTOS, FreeBSD…)

Quotes

"The CHERI architecture's support for fine-grain memory protection and scalable compartmentalization promises to revolutionise our ability to protect personal data and provide strong defences against malware on mobile devices and in the cloud."

Ben Laurie, Director of Security at Google Research

"As noted by the White House in a recent report on a path toward secure and measurable software, hardware support is critical to robust and efficient memory safety. Compiling software to run on CHERI enhanced processors guarantees very strong memory safety that an attacker cannot bypass."

Professor Simon Moore, University of Cambridge

CHERI Alliance

Thank you!

Web: www.cheri-alliance.net