CTSRD

CRASH-worthy Trustworthy Systems Research and Development

CHERI: a research platform deconflating hardware virtualization and protection

Robert N.M. Watson, Peter G. Neumann
Jonathan Woodruff, Jonathan Anderson, Ross Anderson
Nirav Dave, Ben Laurie, Simon W. Moore, Steven J. Murdoch
Robert Norton, Philip Paeps, Michael Roe, Hassen Saidi

RESoLVE'12
London, UK
2 March 2012

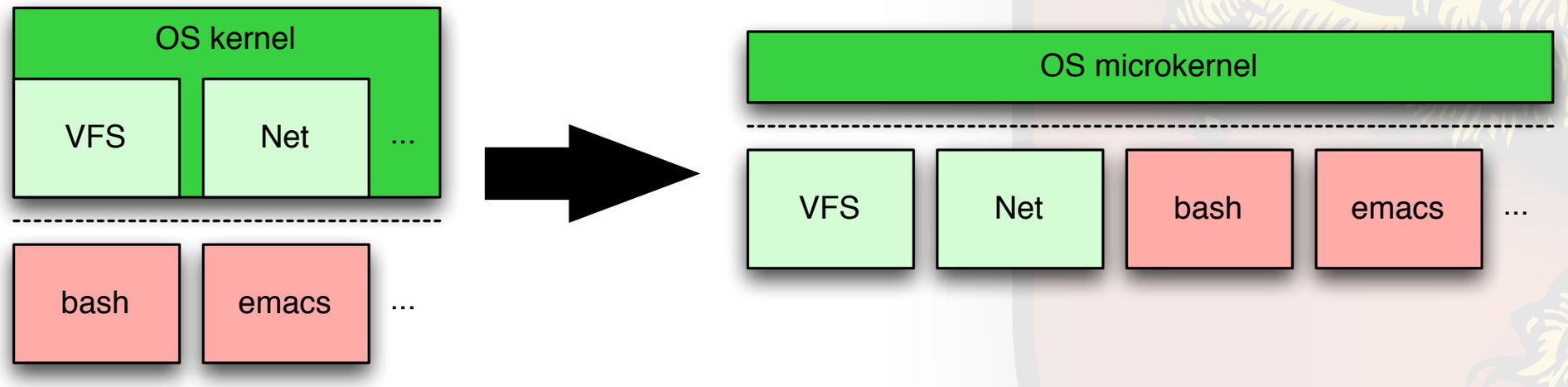
Approved for public release. This research is sponsored by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL), under contract FA8750-10-C-0237. The views, opinions, and/or findings contained in this article/presentation are those of the author/presenter and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.

SRI International

University of Cambridge

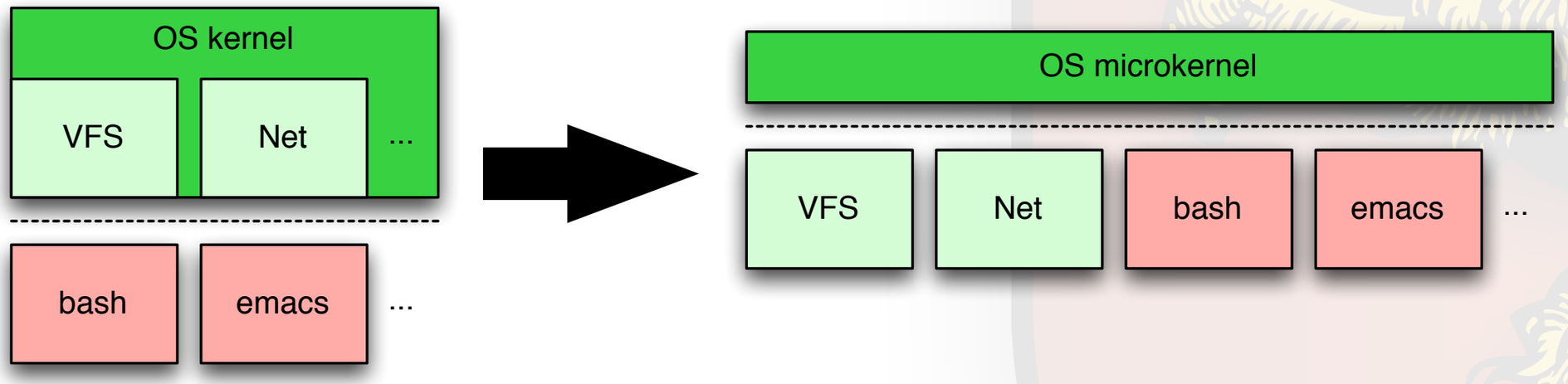
From microkernels to compartmentalisation

1980's

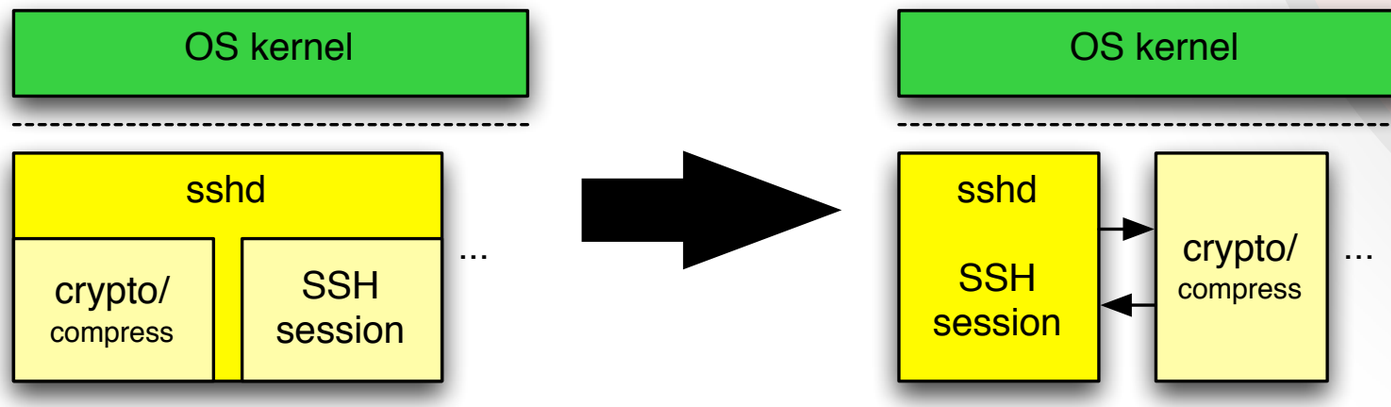


From microkernels to compartmentalisation

1980's

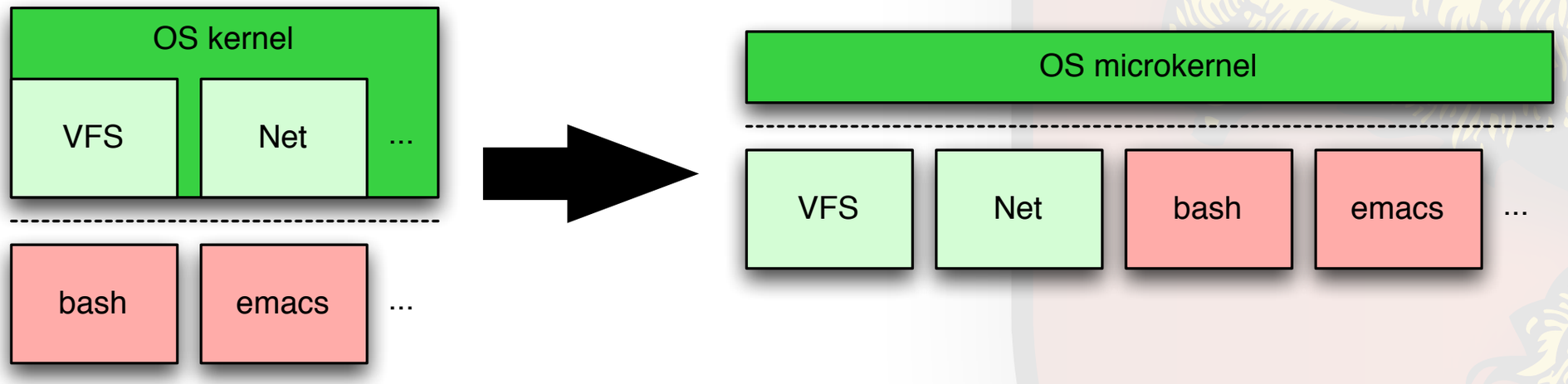


2000's

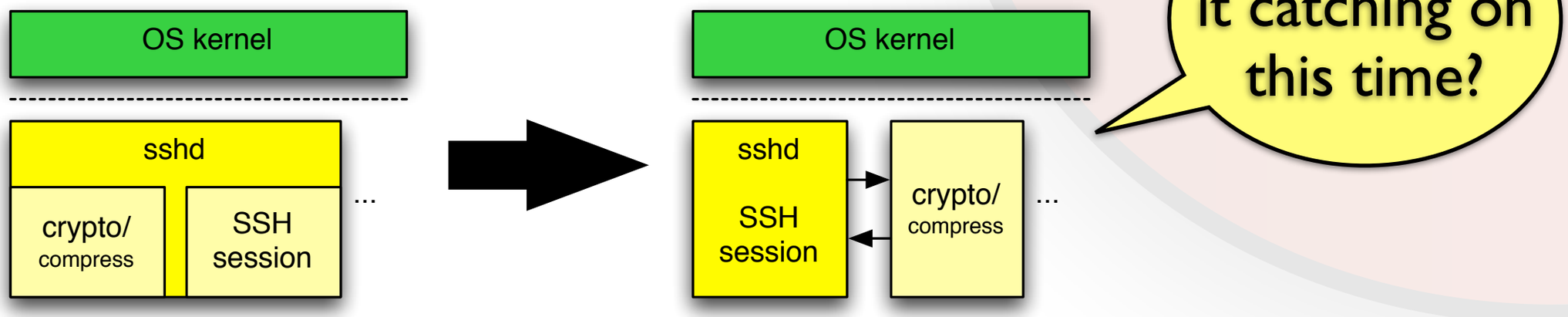


From microkernels to compartmentalisation

1980's

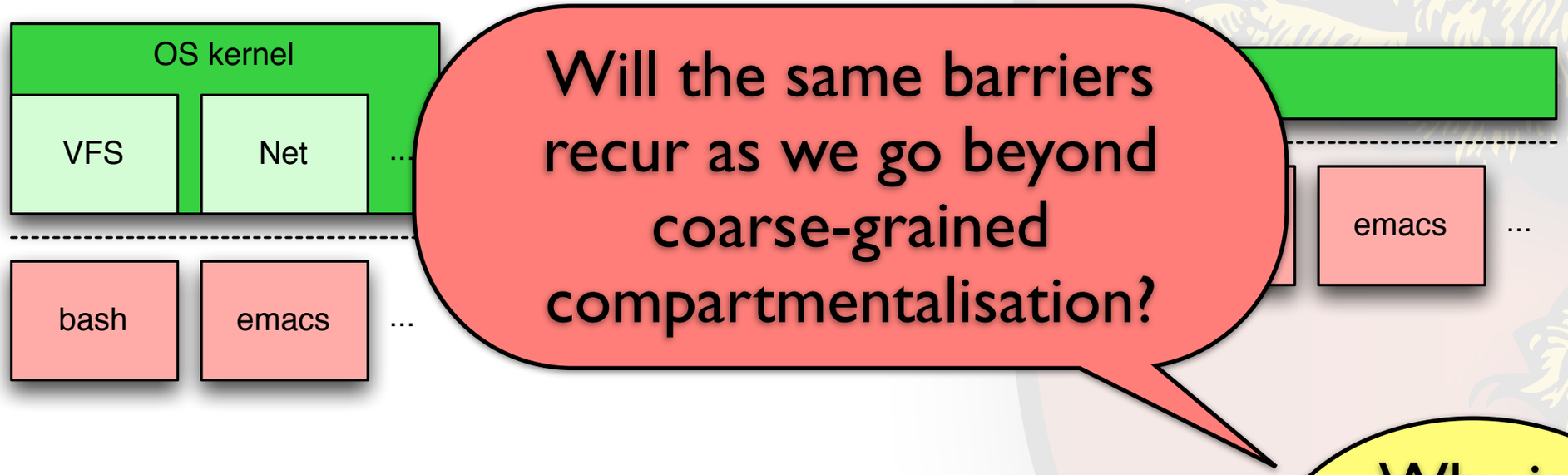


2000's

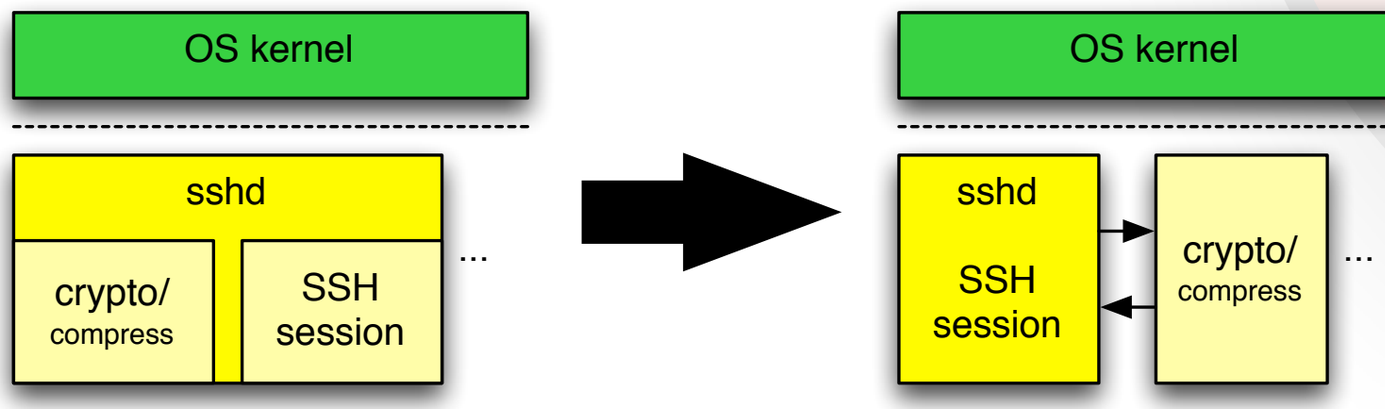


From microkernels to compartmentalisation

1980's



2000's



Capsicum

- Capsicum hybrid capability model: incremental adoption strategy
  - Run current applications, selectively deploy capabilities in TCB, vulnerable libraries and applications
  - Short-term benefits, long-term vision
- Software implementation of the principle of least privilege is neither easily nor efficiently represented in current hardware
- C-language kernels and language runtimes (TCBs) are enormous and unsound -- but amazingly persistent
- Software TCB implementations embody artefacts of security policies rather than design principles

DARPA CRASH

If you could revise the fundamental principles of computer system design to improve security…

…what would you change?

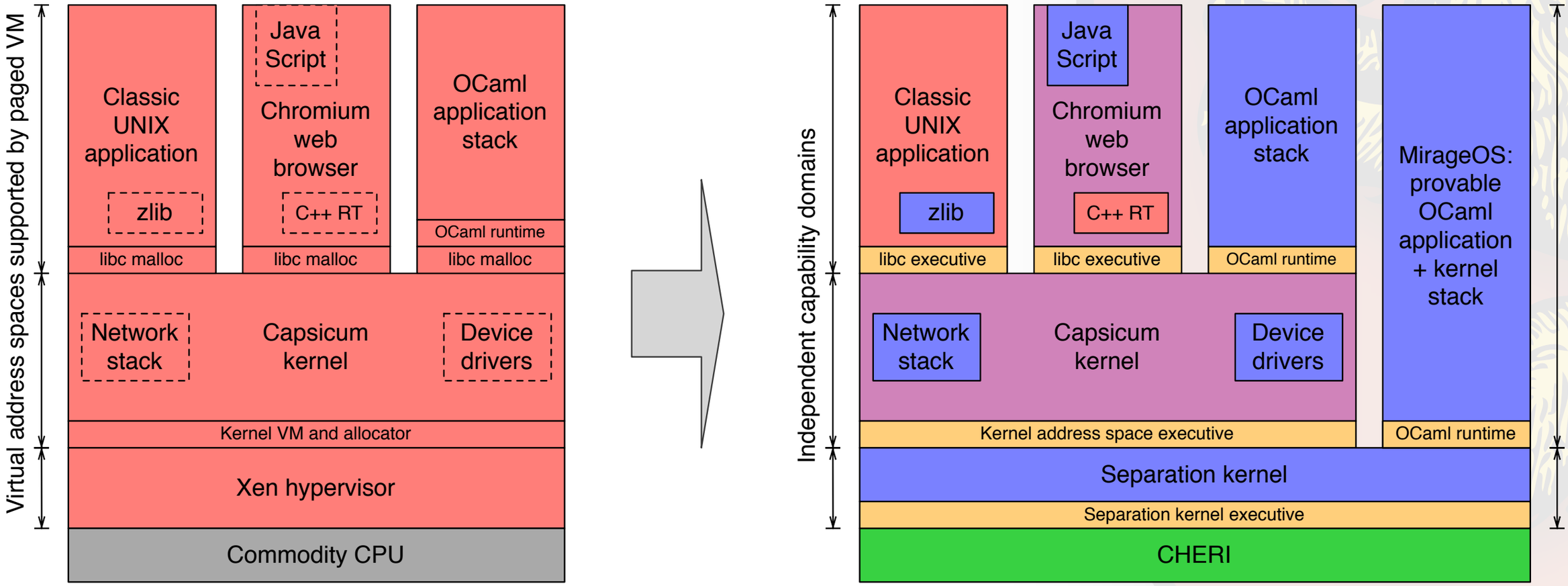
What has changed since current CPU protection models were developed?

- Trend towards **exposing inherent hardware parallelism** to the programmer: software context switching can now be avoided
- Mature translations from type-safe language to **expression-limited byte codes**, e.g., Java, CLR — security not assured, but at least possible
- Pressing **security motivation** for fine-grained software compartmentalisation
- New opportunities for hardware-software research created by **FPGA soft cores**, **open source software** and **mobile computing platforms**

CHERI MIPS

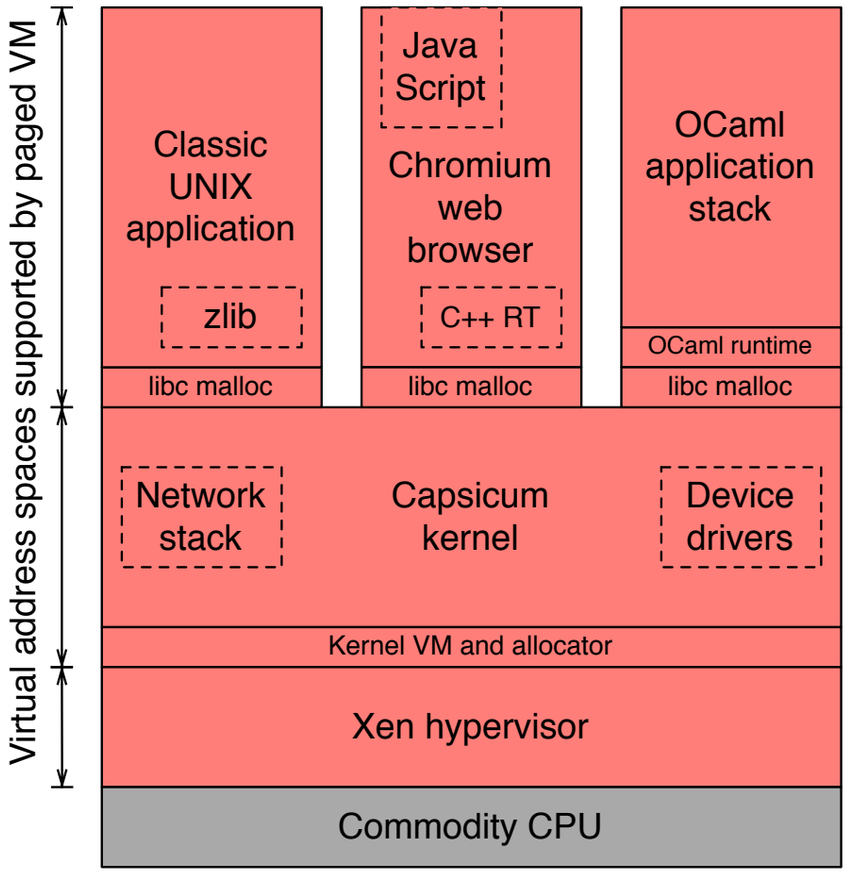
- Transpose ideas from Capsicum into CPUs
  - Capability hardware enhanced RISC instructions
  - Deconflate virtualisation and protection
  - Add fine-grained in-address space protection…
    … but retain the MMU to support VMs and processes
  - Hybrid capability model: current OS, applications
- Experiment with C-language TCBs, vulnerable libraries
  - FreeBSD, LLVM, Apache, Chromium, …
- Experimental questions: hardware or software enforcement? Nature and expression of protection?

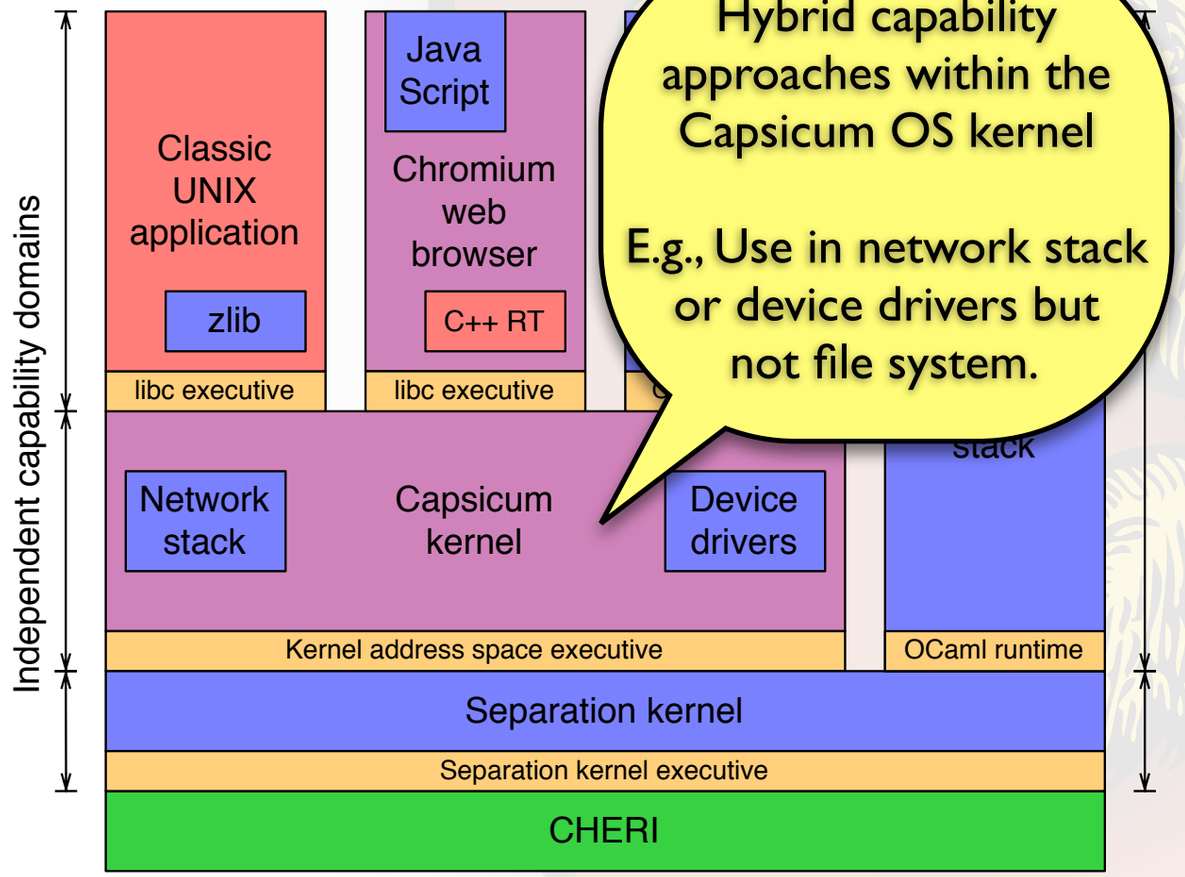
CHERI software architecture



- Legacy application code compiled for general-purpose registers
- Hybrid code blending general-purpose registers and capabilities
- High-assurance capability-only code; stand-alone or "pools of capabilities"
- Per-address space memory management and capability executive

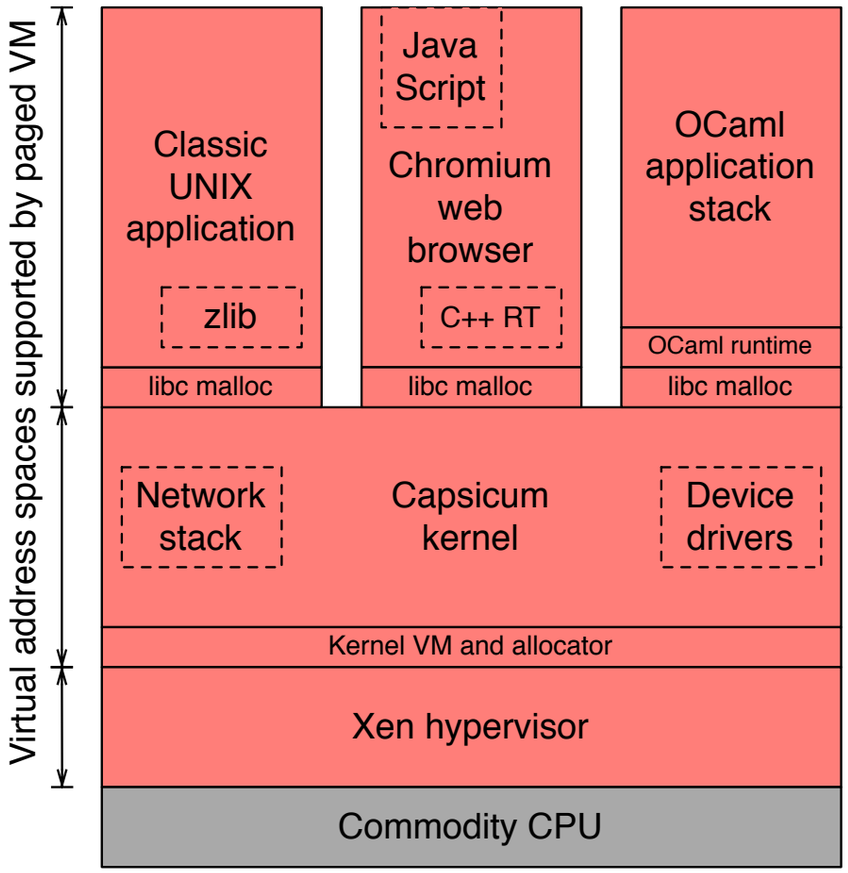
CHERI software architecture

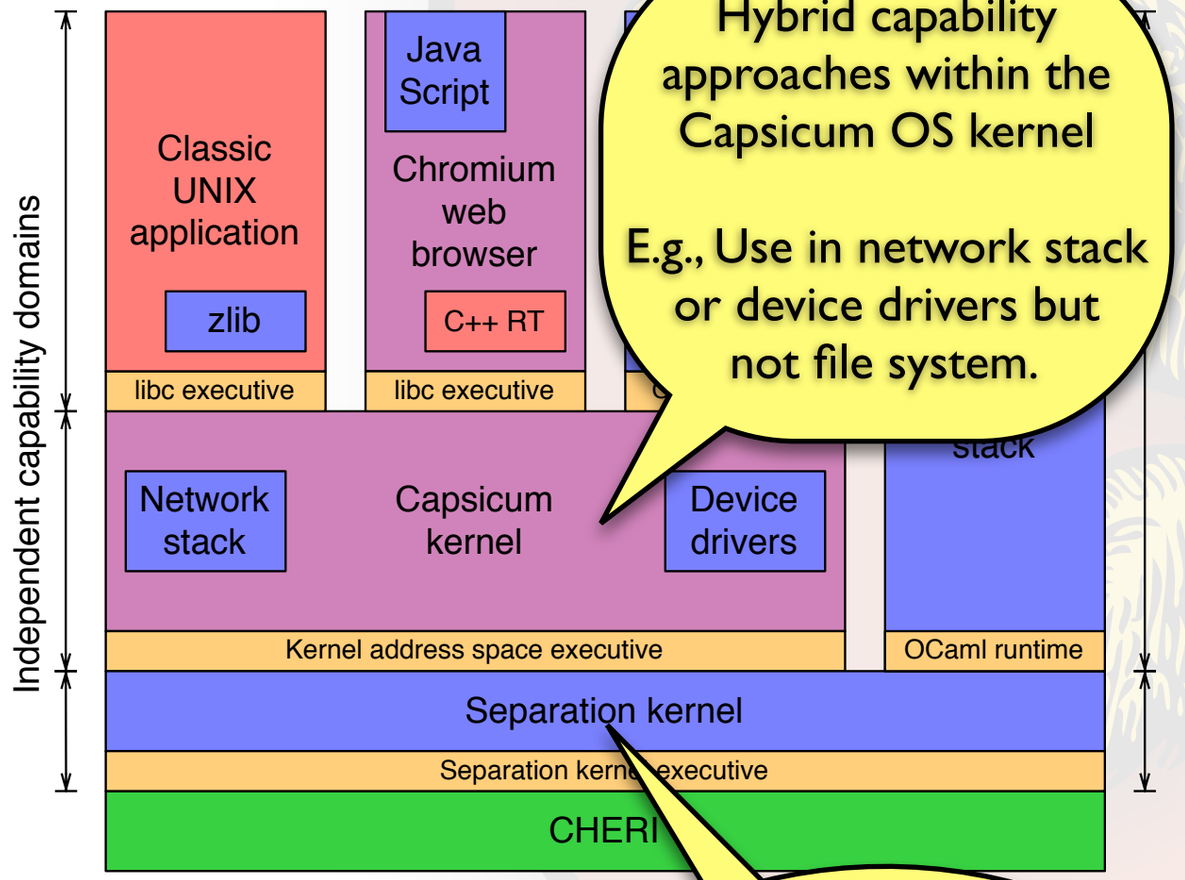




- Legacy application code compiled for general-purpose registers
- Hybrid code blending general-purpose registers and capabilities
- High-assurance capability-only code; stand-alone or "pools of capabilities"
- Per-address space memory management and capability executive

CHERI software architecture





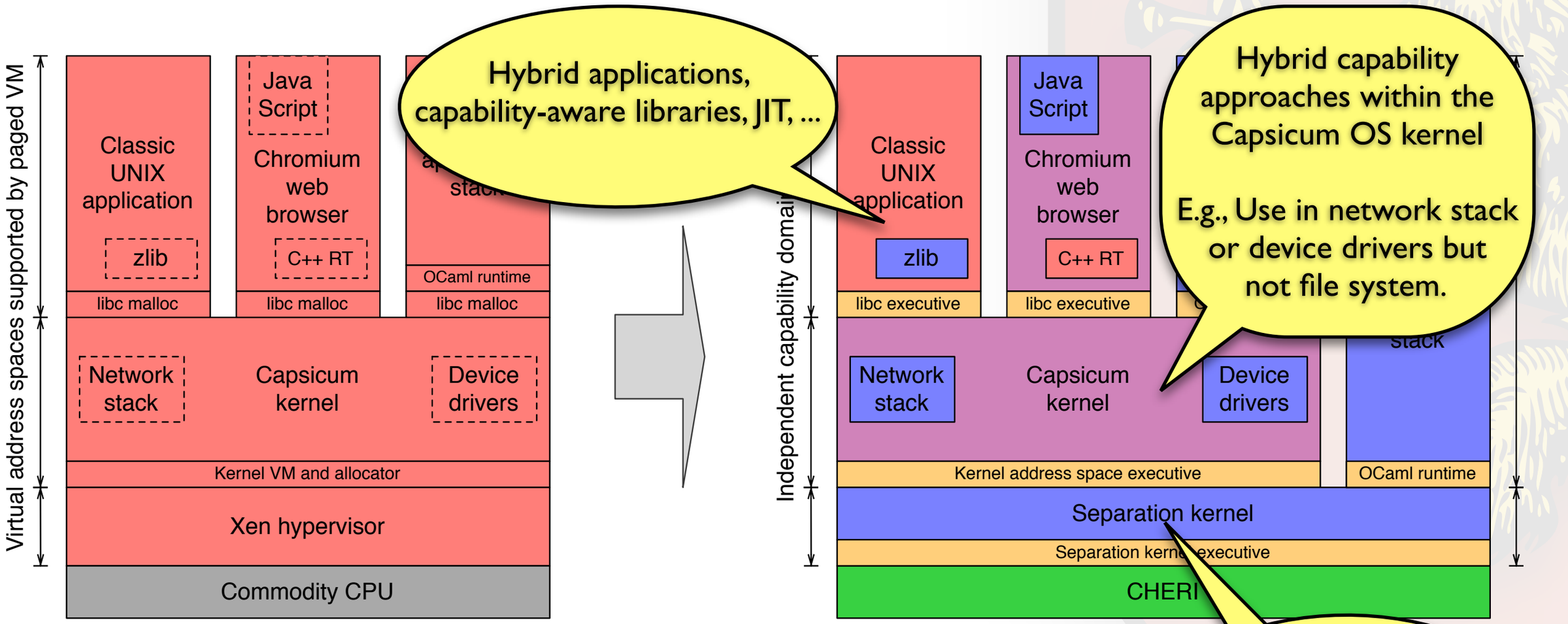
Hybrid capability approaches within the Capsicum OS kernel

E.g., Use in network stack or device drivers but not file system.

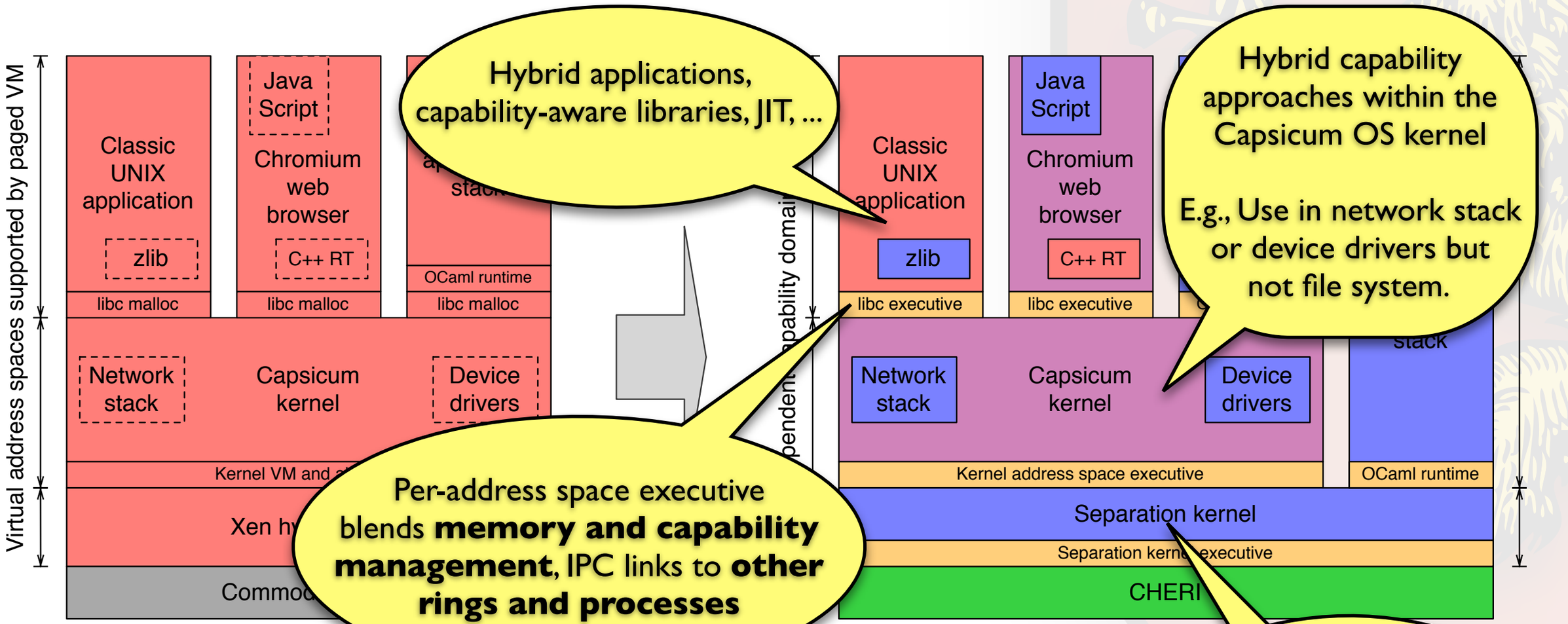
The separation kernel will support both MMU separation from guests **and** capability interfaces to pure capability guests.

- Legacy application code compiled for general-purpose registers
- Hybrid code blending general-purpose registers and capabilities
- High-assurance capability-only code; stand-alone or "pools of capabil
- Per-address space memory management and capability executive

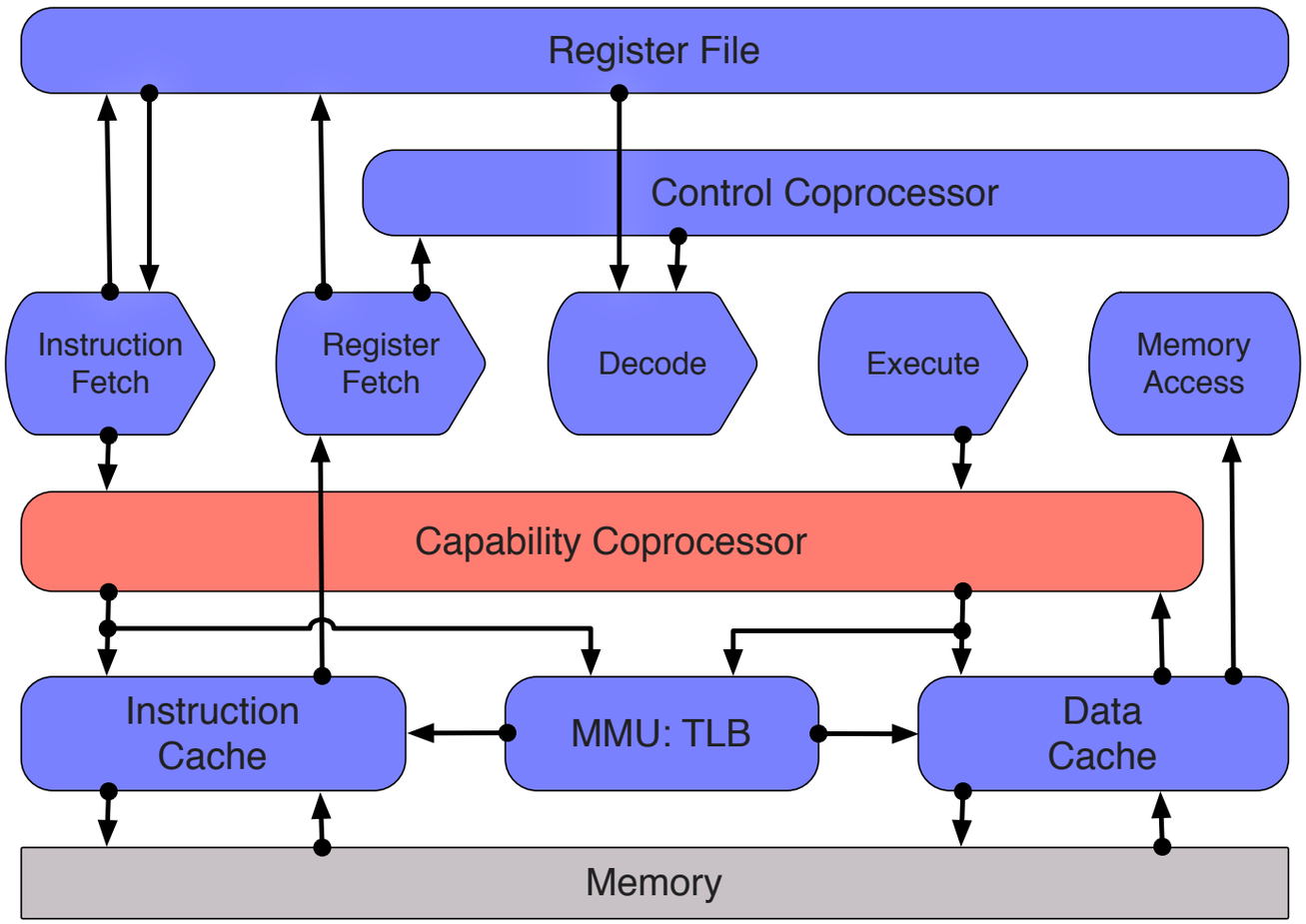
CHERI software architecture



CHERI software architecture

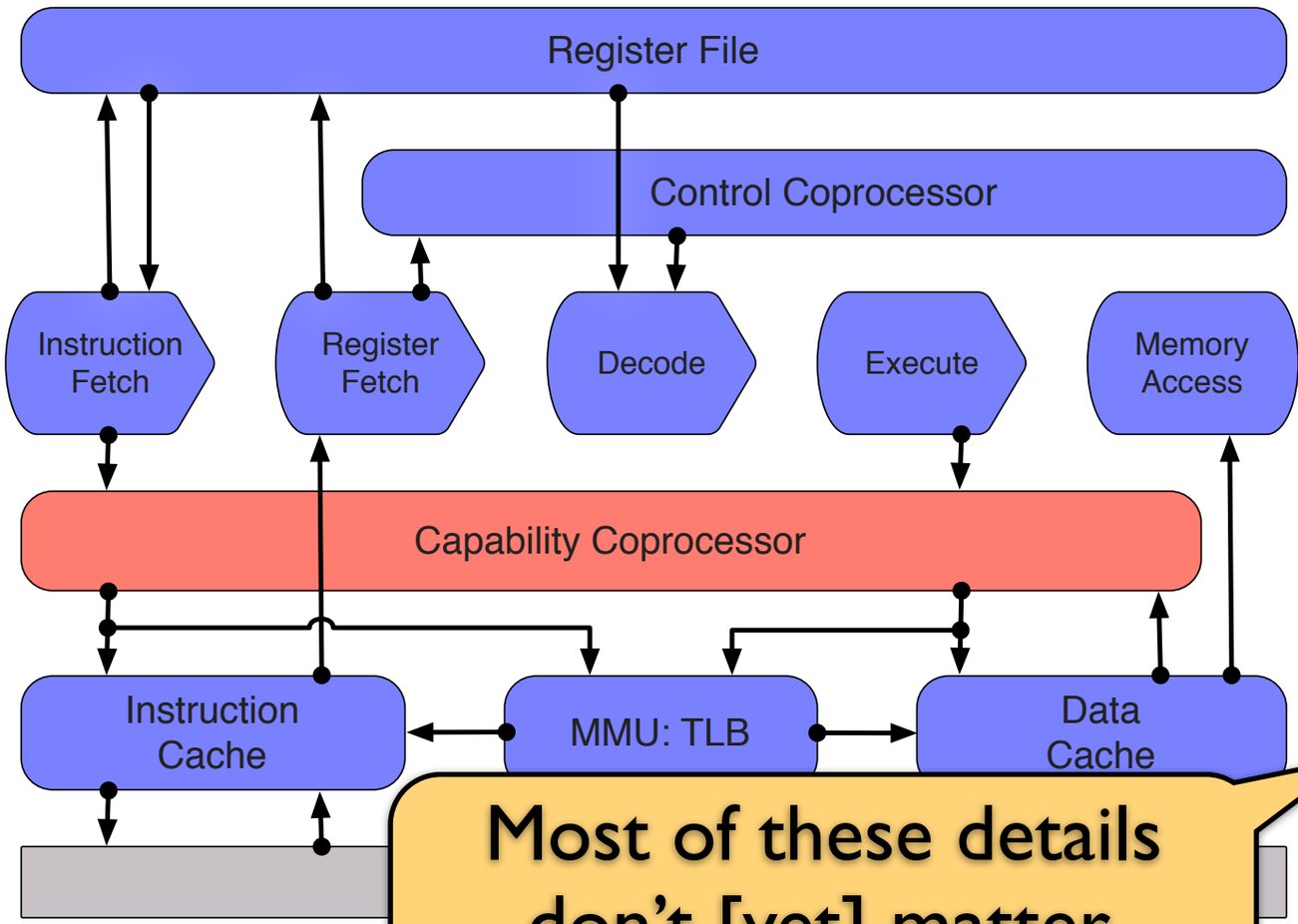


CHERI CPU architecture



- Capability coprocessor
- Capability registers supplement general-purpose registers
  - Describe segments, objects
  - Compiler-managed
  - Unprivileged access
  - “Fat pointers”
- Hybrid operation transforms general-purpose memory accesses
- Object capabilities employ in-development call-gate facility

CHERI CPU architecture



Most of these details don't [yet] matter.

They become parameters for future experiments.

- Capability coprocessor
- Capability registers supplement general-purpose registers
  - Describe segments, objects
  - Compiler-managed
  - Unprivileged access
  - "Fat pointers"
- Hybrid operation transforms general-purpose memory accesses
- Object capabilities employ in-development call-gate facility

SRI International

UNIVERSITY OF CAMBRIDGE

So, you want to do research into the hardware software interface…

…where do you begin?

The hardware-software interface research problem

- Hardware, software, and network protocol researchers work in largely independent silos
  - Treat each others' corpuses as constants for experiments
- But we want to answer **multi-variable** research questions:

➡ What happens as we vary both TLB size and OS strategy?

➡ Was conflation of CPU memory virtualisation and memory protection a mistake?

➡ What are the interactions of energy efficiency optimisation across both hardware and software?

➡ How can a "portable" operating system message passing semantic span a variety of hardware semantics?

BERI

Bluespec Extensible RISC Instructions

Reference applications

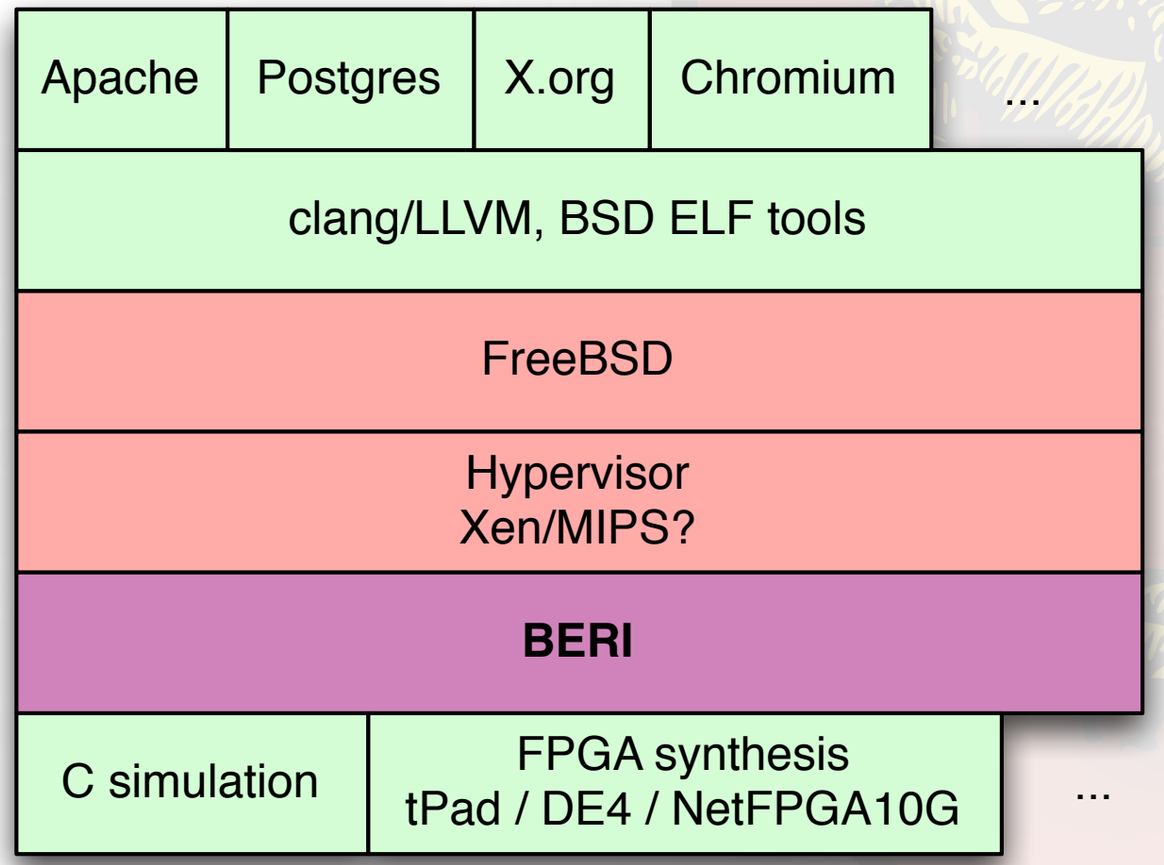
Reference compiler/toolchain

Reference operating system

Reference hypervisor

Hardware research stack

Hardware simulation/
implementation substrates



Complete hardware-software research platform

Apache/BSD-licensed from top to bottom

SRI International

UNIVERSITY OF CAMBRIDGE

BERI status

- 16 months in
  - Soft single-core 64-bit MIPS processor
  - Terasic DE-4, tPad: - Altera FPGA + certain peripherals
  - Uboot boot loader, research Deimos microkernel
- In progress
  - FreeBSD adaptation -- creeping up on single-user mode
  - 64-bit MIPS LLVM backend
  - First research project: CPU capability protection model
- Now starting on...
  - Multithreading, multicore
  - Rackscale memory interconnects
  - Port to NetFPGA 10G platform

Unusual OS port perspective: **fix hardware** rather than work around in software!

Immediate research applications

- Revisit historic RISC assumptions
- Hardware cache strategies vs. OS scheduling
- Exploiting memory locality information for hardware thread thread scheduling
- How should operating systems "portably" span multiple hardware message passing semantics
- Does fine-grained protection belong in hardware or software?
  - Virtualisation vs. protection

CHERI status

- Fleshing out ISA test suite, pipeline fuzzer, etc.
- New "cheri2" in flight to support formal methods
- CHERI adaptations to OS, toolchain
  - FreeBSD port mid-stride
  - LLVM work beginning
  - Developing ABIs, application models
  - Pondering C language extensions, pilot components/applications
- Preparing to enter experimental phase
  - Side-by-side hardware and software implementations of semantics
  - Comparisons between conventional and capability-based models

Collaborative project

Cambridge

Architecture: Jonathan Woodruff, Simon Moore, Greg Chadwick, Alan Mujumdar, Robert Norton, *Wojciech Koszek*

Security: Jon Anderson, Ross Anderson, Ben Laurie, Steven Murdoch, Philip Paeps, Michael Roe, *Ilias Marinos*

NetOS: Anil Madhavapeddy, Andrew Moore, Steven Hand, Muhammad Shahbaz, *Will Morland*

SRI

Systems, Formal Methods: Peter Neumann, Nirav Dave, Hassen Saidi, Rance DeLong, John Rushby, Pat Lincoln, Natarajan Shankar

Conclusion

- Four-year "cross-disciplinary" collaborative project
- BERI: Bluespec experimental RISC implementation
  - Research platform for the hardware-software interface
- CHERI: Capability hardware enhanced RISC instructions
  - Have we undesirably conflated protection and virtualisation?
  - Does fine-grained protection belong in hardware or software?
- Support from DARPA, Google
- http://www.cl.cam.ac.uk/research/security/ctsrd/